UNIVERSITÀ DEGLI STUDI DI
PADOVA

DIPARTIMENTO DI INGEGNERIA
DELL'INFORMAZIONE
CORSO DI LAUREA IN INGEGNERIA INFORMATICA

REALIZZAZIONE DI APPLICAZIONI INFORMATICHE PER L'ACCESSO ALLE FUNZIONALITÀ DI UFFICIO IN MOBILITÀ

Laureando
Luca Gasparini

Relatore
Michele Moro

Anno accademico 2009/2010

SOMMARIO

La tesi si propone di illustrare il lavoro svolto durante lo stage presso l'azienda I.C.H.I. SRL di Castelfranco Veneto nei mesi di Marzo, Aprile, Maggio, Giugno e Luglio 2010. Lo scopo di questa attività di tirocinio è stato quello di studiare le diverse tecnologie attualmente a disposizione per la realizzazione di strumenti software per l'ufficio utilizzabili in mobilità. In particolare mi è stato proposto di realizzare un'interfaccia per un sistema di gestione per avvocati, già realizzato dall'Azienda ospitante, con il requisito che esso sia accessibile da un qualsiasi dispositivo mobile come uno smartphone, un palmare o un notebook. Per fare ciò si è deciso di sviluppare un'applicazione Web, in quanto, l'applicazione risultante, sarebbe stata accessibile da un qualsiasi dispositivo dotato di un browser Web.

Dopo aver valutato diverse alternative, vista la precedente esperienza dell'azienda con gli strumenti di sviluppo della piattaforma Microsoft .NET si è deciso di realizzare la Web-application utilizzando la tecnologia ASP.NET facente parte appunto della suddetta piattaforma. Per assicurare un certo livello di segretezza dei dati trasmessi si è inoltre deciso di utilizzare il protocollo https che permette lo scambio di dati sicuri tra client e server. Durante il periodo di stage è stata sviluppata una demo che ha permesso di valutare la fattibilità del progetto e di illustrarne una parte delle funzionalità.

INDICE

1 INTRODUZIONE

Negli ultimi anni abbiamo assistito ad una notevole evoluzione del Web grazie all'introduzione di nuove tecnologie che hanno permesso di creare pagine Web sempre più interattive, tanto da poter parlare di vere e proprie applicazioni Web. La necessità di realizzare un'applicazione accessibile da dispositivi molto diversi tra loro (notebook, smartphone, palmari) ci ha indirizzato verso lo sviluppo di un'applicazione Web, in quanto questa risulterà eseguibile in un qualsiasi dispositivo dotato di un browser Web. Dopo aver esaminato il problema e valutato l'effettiva possibilità di realizzare l'applicazione si è passati alla scelta della tecnologia di sviluppo da utilizzare. Vista l'esperienza acquisita dall'azienda ospitante nell'utilizzo della piattaforma di sviluppo Microsoft.NET, si è deciso di sviluppare l'applicazione Web utilizzando la tecnologia Microsoft ASP.NET, che permette di creare, utilizzando il modello di programmazione e il framework Microsoft .NET, Web Service e applicazioni web. Il flusso di dati relativo all'applicazione Web normalmente viene trasmesso utilizzando il protocollo http che non offre alcuna garanzia in termini di sicurezza e segretezza dei dati. Per questo si è deciso di utilizzare il protocollo https che combina http con la tecnologia SSL/TLS rendendolo più sicuro.

Nella realizzazione del progetto sono stati utilizzati i seguenti strumenti:

- ASP.NET (linguaggio C#) .NET Framework 3.5
- SQL Server 2005
- OpenSSL

.Net è un'ambiente di lavoro creato da Microsoft per gli sviluppatori, affinché possano creare applicazioni che risolvono i bisogni degli utenti connessi ad internet.

Lo scopo principale della tecnologia ASP.NET è quello di creare applicazioni Web alle quali possono accedere vari dispositivi, con funzionalità simili a quelle di un'applicazione Windows. Utilizzando il framework .NET è possibile scrivere applicazioni in diversi linguaggi di programmazione:

- C#
- Jscript.NET
- J#
- VB.NET

e anche altri linguaggi creati da terzi. Infatti il codice scritto in uno qualsiasi di questi linguaggi viene compilato in un linguaggio intermedio comune (CIL) che viene poi tradotto e assemblato a runtime ed eseguito dal Common Language Runtime (CLR) . Il ruolo del CLR è molto simile a quello della Java Virtual Machine, ma , compilando nativamente tutto il codice, garantisce migliori prestazioni rispetto ad essa.

Il linguaggio di programmazione C# (si legge “C sharp”) deriva dai linguaggi C e C++, ma si tratta di un linguaggio moderno, semplice, interamente ad oggetti e dotato di tipi più sicuri. Esso può essere considerato il linguaggio di programmazione per eccellenza del Framework .NET in quanto, diversamente dagli altri linguaggi, esso è nato espressamente per la nuova piattaforma. In questo senso, è significativo il fatto che Microsoft stessa si sia servita di C# per scrivere gran parte delle librerie di .NET; uno degli slogan che hanno accompagnato C# fin dalla sua nascita lo presenta come un “linguaggio facile come Java ma potente come C++”.

Per quanto riguarda l'IDE, è stato utilizzato Visual Studio .NET, un ambiente di sviluppo integrato sviluppato da Microsoft che supporta diversi tipi di linguaggio tra cui C++, C#, J#, Visual Basic .NET e ASP.NET e che permette

la realizzazione di applicazioni, siti web, applicazioni web e servizi web. È inoltre un RAD (Rapid Application Development), ovvero un'applicazione atta ad aumentare la produttività del programmatore con mezzi come l'intellisense o un designer visuale delle forms. La release utilizzata è Visual Studio .NET 2008 che utilizza il framework .NET 3.5 e, rispetto alle versioni precedenti, introduce LINQ che permette di effettuare interrogazioni simili a quelle in linguaggio SQL su normali oggetti e termina il supporto al linguaggio J#.

Microsoft SQL Server 2005 Express Edition è un Data Base Management System sviluppato da Microsoft che, a differenza di SQL Server 2005, è disponibile gratuitamente. Esso supporta il modello di programmazione completo di SQL Server 2005 inclusi i transact-SQL, stored procedure, viste, trigger, SQL Server CLR Integration e il tipo di dati XML.

In generale le applicazioni web accedono alle origini dati per la memorizzazione e il recupero di dati dinamici: è possibile scrivere il codice per l'accesso ai dati con ASP.NET utilizzando le classi dello spazio dei nomi System.Data, comunemente denominato ADO.NET, e dello spazio dei nomi System.XML.

OpenSSL è una implementazione open source dei protocolli SSL/TLS per la certificazione e la comunicazione cifrata. OpenSSL si compone di alcune librerie che permettono di incorporare le funzionalità dei protocolli SSL/TLS all'interno di altri programmi, e di una serie di programmi di utilità per la gestione delle chiavi e dei certificati, arrivando eventualmente anche alla gestione di un'autorità di certificazione. Con OpenSSL si è quindi in grado di creare e gestire certificati che serviranno per far comunicare tra loro in modo sicuro i server e i client.

Lo scopo principale del lavoro svolto durante il periodo di stage consisteva

nello studiare le tecnologie a disposizione per la realizzazione di applicativi che permettano di accedere alle funzionalità di un ufficio in mobilità. In particolare ci si è posti l'obiettivo di realizzare una demo di un'interfaccia per un software di gestione di uno studio legale già sviluppato dall'azienda ospitante, in modo da capire le potenzialità e i limiti dell'utilizzo di queste tecnologie. Con lo sviluppo della demo ci si aspetta di ottenere un'applicazione Web, che possa essere eseguita su diversi tipi di dispositivi e che sia il più simile possibile, per quanto riguarda l'aspetto e le funzionalità, alla corrispondente applicazione desktop realizzata precedentemente, tenendo conto dei limiti imposti dalle ridotte risorse disponibili nel client e dalle limitazioni dovute alla banda.

2 MICROSOFT .NET

Il framework Microsoft .NET è l'infrastruttura che costituisce la nuova piattaforma creata da Microsoft per lo sviluppo di applicazioni component-based, n-tier, per internet, o per le classiche applicazioni desktop. La prima versione di .NET è stata rilasciata nel 2002. La sua caratteristica peculiare è di essere indipendente dalla versione operativa di Windows su cui è installata, e di includere molte funzionalità progettate espressamente per integrarsi in ambiente internet e garantire il massimo grado di sicurezza e integrità dei dati.

Le caratteristiche principali della piattaforma .NET sono le seguenti:

Interoperabilità*:* è richiesta la compatibilità tra nuove e vecchie applicazioni, quindi deve essere possibile, accedere a funzionalità implementate in applicazioni che vengono eseguite all'esterno del ambiente .NET. È possibile accedere ai componenti COM utilizzando i namespaces System.Runtime.InteropServices and System.EnterpriseServices namespaces.

Ambiente di esecuzione comune*:* il CLR (Common Language Runtime) è la macchina virtuale del framework .NET. Tutti i programmi sono eseguiti sotto la supervisione del CLR che garantisce alcune caratteristiche nell'ambito della gestione della memoria, della sicurezza, e delle eccezioni.

Indipendenza dal linguaggio utilizzato: il framework .NET introduce un sistema di tipi comuni (CTS) , che definisce tutti i diversi tipi e costrutti supportati dal CLR e come essi possono interagire tra di loro. Grazie al CTS il .NET framework supporta lo scambio di tipi e istanze tra librerie scritte usando linguaggi .NET differenti.

Base Class Library: è una libreria, parte della Framework Class Library,

accessibile da tutti i linguaggi della piattaforma .NET. Questa libreria fornisce classi che implementano funzionalità di base riguardanti la gestione dei file, la manipolazione grafica, l'interazione con i database, manipolazione dei dati XML e molto altro.

Sicurezza: .NET è stato pensato per affrontare alcune delle vulnerabilità, come il buffer overflow, che vengono sfruttare per scrivere software dannoso. Inoltre, .NET fornisce un modello comune di sicurezza per tutte le applicazioni.

Portabilità: Il Framework .NET è stato progettare per essere completamnte indipendente dalla piattaforma in cui viene eseguito. Questo permette di eseguire un'applicazione scritta per girare sull'ambiente .NET in una qualsiasi piattaforma per la quale il framework è stato implementato. Microsoft non ha implementato il suddetto framework per nessuna piattaforma diversa da Microsoft Windows ma ha reso disponibili le specifiche del CLI cosicché sia possibile per terze parti implementarlo su una qualsiasi piattaforma.

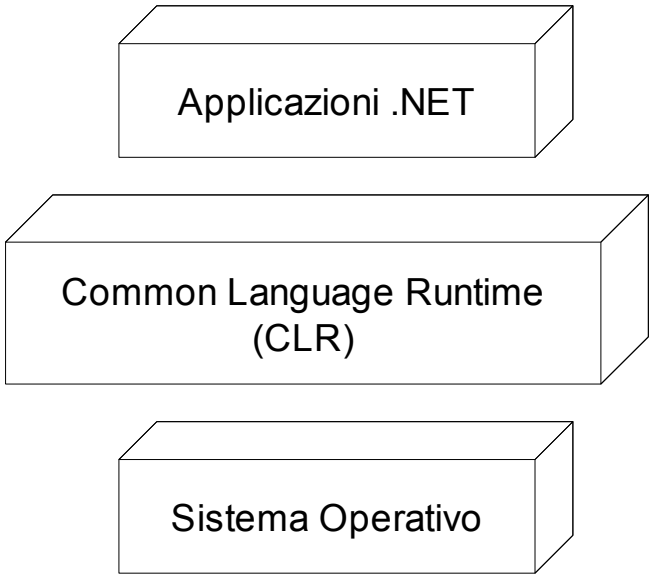
2.1 FRAMEWORK .NET

Il framework Microsoft.NET copre ogni aspetto relativo allo sviluppo del software schermando totalmente il sistema operativo su cui ci si trova. Il framework offre allo sviluppatore un ambiente completamente gestito per lo sviluppo di applicazioni occupandosi di ogni aspetto, dalla fase di esecuzione e gestione delle risorse e della memoria, all'accesso ai dati, alla gestione dell'interfaccia. Il framework .NET si compone di tre parti principali:

- compilatori per i principali linguaggi supportati da Microsoft;
- ambiente di esecuzione Common Language Runtime (CLR);
- libreria di classi.

Il principale concetto del framework è il Common Language Runtime: è un livello del framework posto sopra il sistema operativo e che gestisce l'esecuzione delle applicazioni .NET. I programmi scritti in .NET non comunicano direttamente con il sistema operativo ma attraverso il CLR. Esso è responsabile dell'esecuzione vera e propria delle applicazioni, assicura che vengano rispettate tutte le dipendenze, gestisce la memoria, la sicurezza, l'integrazione del linguaggio e cosi via. Il runtime fornisce numerosi servizi che consentono di semplificare la stesura del codice, la distribuzione dell'applicazione e di migliorare l'affidabilità della stessa. Il Common Language Runtime rappresenta un motore di esecuzione a elevate prestazioni. Il codice cui il runtime si riferisce e la cui esecuzione è gestita da esso, è detto codice gestito (managed code). Codice macchina insicuro (perché scavalca il runtime) può essere generato dai compilatori, ed in questo caso si parla di codice non gestito (unmanaged code). L'accesso alle Garbage Collection è possibile solo attraverso il codice

gestito. La responsabilità per attività quali la creazione di oggetti, l'esecuzione di chiamate a metodi e cosi via, è demandata al Common Language Runtime che consente di fornire servizi aggiuntivi al codice in esecuzione. Il codice gestito ha accesso al Common Language Runtime attraverso il quale può avvantaggiarsi delle caratteristiche della piattaforma (integrazione multi-linguaggio, gestione delle eccezioni, sicurezza, gestione delle versioni, ecc.).



Il Common Language Runtime è composto da cinque componenti che sono:

- CTS – Common Type System: il runtime utillizza un sistema di tipi unificato in grado di esprimere la semantica dei moderni linguaggi di programmazione. Tale sistema definisce un insieme standard di tipi di dato e di regole necessarie per la realizzazione di nuovi tipi. Il runtime è in grado di capire come creare ed eseguire tali tipi. I compilatori del .NET Framework utilizzano i servizi del runtime per definire i tipi di dato, gestire gli oggetti ed eseguire chiamate a

metodi invece di utilizzare i metodi specifici dello strumento o del linguaggio. L'utilizzo di un sistema di tipi unificato offre come risultato una profonda integrazione tra i linguaggi. Il codice scritto in un linguaggio può ereditare l'implementazione da classi scritte in un altro linguaggio; le eccezioni possono essere sollevate dal codice scritto in un linguaggio e gestite da codice scritto in un altro, e operazioni come il debugging e profiling operano in modo trasparente indipendentemente dal linguaggio utilizzato per scrivere il codice. Ciò significa che gli sviluppatori non hanno più bisogno di creare versioni differenti delle proprie librerie per ogni linguaggio di programmazione o compilatore, e che, per quanto riguarda le librerie di classe, non sono più limitati a quelle sviluppate per il linguaggio di programmazione utilizzato.

- CLS – Common Language Specification: Il CLS definisce un sottoinsieme del Common Type System al quale tutti i fornitori di librerie di classi e progettisti di linguaggi che puntano al CLR, devono aderire. Il CLS è una serie di regole che si applicano per generare gli assembly. Se un componente scritto in un linguaggio (ad esempio C#) dovrà essere utilizzato da un altro linguaggio (ad esempio VB.NET), allora chi scrive il componente dovrà aderire ai tipi e alle strutture definite dal CLS. I CLS framework devono sottostare a una serie di regole, tra le quali:
  - evitare l'uso di nomi utilizzati comunemente come parole chiave nei linguaggi di programmazione;
  - non dovrebbero permettere all'utente di costruire tipi nidificati;
  - si assume che le implementazioni dei metodi con lo stesso nome e firme in differenti interfacce siano indipendenti.

- CIL – Common Intermediate Language L'implementazione nel .NET Framework del Common Intermediate Language è chiamata Microsoft Intermediate Language (MSIL). Tutti i compilatori che aderiscono alla struttura del CLR devono generare una rappresentazione intermedia del codice, indipendente dalla CPU, chiamata Common Intermediate Language. Il runtime utilizza questo linguaggio intermedio per generare codice nativo oppure viene eseguito al volo mediante la compilazione Just In Time. Il CIL si pone a un livello molto più alto della maggior parte dei linguaggi macchina, avendo istruzioni per il caricamento, la memorizzazione e l'inizializzazione dei dati, per richiamare metodi da oggetti e molte istruzioni di tipo convenzionale per le operazioni aritmetiche e logiche, il controllo di flusso e l'accesso diretto alla memoria. Il CIL possiede inoltre istruzioni per elevare e intercettare eccezioni per la gestione degli errori, simili a quelle Java di tipo “try/catch”. Questo formato intermedio presenta contemporaneamente affinità e grandi differenze rispetto al tradizionale linguaggio Assembly, similmente al quale può operare sui dati con istruzioni di tipo “push” e “pop” e spostarli in registri; ma diversamente dallo stesso non fa riferimento a nessuna particolare piattaforma hardware. Uno dei principali vantaggi di questa soluzione è che permette al CLR di verificare durante la compilazione che il codice gestito sia completamente “Type Safe”, ovvero conforme alle specifiche di programmazione e ai tipi di dati previsti dal runtime. Durante questa verifica il CLR controlla ad esempio l'uso corretto dei puntatori e si assicura che non siano presenti conversioni tra tipi non consentite. Proprio come Java, prima che il codice gestito sia eseguito, l'IL è convertito al volo in codice specifico per la CPU da un compilatore JIT, oppure

compilato in codice nativo durante l'installazione. La piattaforma runtime fornisce uno o più compilatori a seconda del numero di piattaforme che deve supportare: nonostante l'attuale limitazione ad ambienti Microsoft, questo garantisce una buona indipendenza dall'hardware potendo funzionare ad esempio con Windows CE su piattaforma non Intel x86 o sull'architettura Win64. Quando un compilatore conforme al CLS genera il linguaggio intermedio, produce anche metadati che descrivono i tipi specifici appartenenti al Common Language Types (CLT) utilizzati nel codice, comprendente la definizione di ogni tipo, le firme per ogni membro del tipo, i membri ai quali il codice fa riferimento e gli altri dati che il runtime usa durante l'esecuzione. Il MSIL e i metadati sono contenuti in un file Portable Executable (PE), un'estensione del formato Microsoft Portable Executable e simile al Common Object File Format utilizzato nel mondo Unix per gli eseguibili. All'utente i PE appaiono come familiari file .DLL e .EXE. Il formato dei file permette di ospitare sia il codice IL che il codice nativo, i metadati e un “pattern signature” che permette al sistema operativo di riconoscere le “immagini” (nel senso di unica porzione contigua di codice) del Common Language Runtime. La presenza dei metadati nei file eseguibili permette ai componenti di essere autodescrittivi, eliminando di fatto la necessità di librerie dei tipi aggiuntive o dei Interface Definition Language (IDL) usate in DCOM e CORBA. Il runtime localizza ed estrae i metadati del file quando è necessario durante l'esecuzione del codice.

- JIT – Just In Time Compiler: prima che l'Intermediate Languagage possa essere eseguito deve essere convertito dal compilatore Just In Time di .NET Framework in codice nativo, che è specifico della CPU e

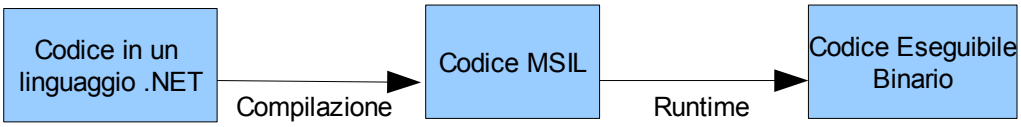
funziona sulla stessa architettura sulla quale il compilatore JIT stesso sta funzionando. I progettisti Microsoft insistono sul fatto che il runtime non interpreta mai nessun linguaggio, ma esegue sempre la conversione e l'esecuzione di codice nativo, persino per i linguaggi di script come VBScript, con evidente vantaggio sulle prestazioni. Il principio di funzionamento che è dietro i compilatori JIT è quello per il quale alcune parti di codice di un programma possono non essere mai chiamate in causa durante l'esecuzione di un programma; quindi piuttosto che sprecare tempo e memoria per convertire tutto il CIL di un file Portable Executable in codice nativo, il JIT converte l'IL al bisogno durante l'esecuzione e memorizza il codice nativo risultante per renderlo disponibile per le chiamate successive. Il loader crea e allega uno stub, un componente software necessario ad eseguire una Remote Procedure Call (RPC), ad ogni metodo del tipo quando il tipo è caricato; alla chiamata iniziale del metodo, lo stub passa il controllo al compilatore JIT il quale converte l'IL di quel metodo in codice nativo e modifica lo stub per dirigere l'esecuzione alla locazione del codice nativo. Le chiamate successive al metodo già compilato dal JIT procedono direttamente verso il codice nativo generato precedentemente, riducendo il tempo necessario alle successive compilazioni ed esecuzioni del programma da parte del compilatore JIT. Il codice cosi compilato (sia mediante JIT che direttamente in forma nativa al momento dell'installazione) deve sottoporsi a un processo di verifica che esamina l'Intermediate Language e i metadati per stabilire che siano “Type Safe” ovvero che accedano esclusivamente alle locazioni di memoria autorizzate, che le identità siano verificate e che i riferimenti ai tipi di dato siano compatibili con i tipi stessi. Durante il processo di verifica, il codice IL

è esaminato in modo da confermare che acceda alle locazioni di memoria e richiami i metodi solo attraverso tipi definiti in maniera corretta e sicuri. Questa caratteristica offre uno strato di protezione automatico dagli errori di programmazione. A causa di limitazioni progettuali di alcuni linguaggi di programmazione come il C, i compilatori di questi linguaggi possono non essere in grado di produrre codice di tipo sicuro e verificabile, quindi questo codice può essere eseguito solo in aree di sicurezza. Sono previsti due tipi di compilatori JIT, quello normale e la versione ridotta economy. Il compilatore normale esamina l'IL di un metodo e lo converte in codice nativo ottimizzato per la piattaforma esattamente come fa un tradizionale compilatore C/C++. Il compilatore economy invece, è stato progettato per l'utilizzo su quelle macchine per le quali il costo per l'uso della memoria e dei cicli di CPU è elevato (come sistemi embedded basati su Windows CE). Il compilatore economy semplicemente rimpiazza ogni istruzione MISL con la sua controparte nativa. Come si può facilmente immaginare questo tipo di compilazione molto più rapida di quella standard, tuttavia il codice prodotto è molto meno efficiente in quanto non soggetto ad ottimizzazione per la specifica piattaforma. Tuttavia questo tipo di codice risulta comunque più efficiente di codice interpretato. Il compilatore economy richiede meno memoria per funzionare, (quindi sarà preferito per i dispositivi portatili) grazie ad una maggiore semplicità progettuale e a meccanismi come il Code Pitching che permette al CLR di eliminare dalla memoria il codice nativo dei metodi non utilizzati. Se un metodo non viene utilizzato per un po' di tempo, il runtime preleva il blocco di codice nativo completo la volta successiva che il metodo verrà invocato.

Probabilmente a un confronto diretto il codice compilato Just In Time con il compilatore standard risulterà comunque più lento del tradizionale codice non gestito, proveniente ad esempio da un compilatore C/C++. Tuttavia per sua stessa natura il compilatore Just In Time lavora direttamente sulla macchina di destinazione del programma e pochi istanti prima di eseguire il programma stesso. Questo fornisce al JIT dati sull'ambiente di destinazione che nessun compilatore tradizionale potrà mai avere permettendo un elevato grado di ottimizzazione. Ad esempio il compilatore può rilevare che la piattaforma di runtime dispone di un processore Pentium IV e compilare il codice con le specifiche istruzioni della CPU, mentre con una procedura tradizionale questa scelta è eseguita in fase di sviluppo del software e spesso gestita con prudenza per garantire la più elevata compatibilità (in alternativa sono fornite più versioni dello stesso programma ottimizzate per le diverse CPU). Anche l'esatta conoscenza dello stato della memoria e dei registri dell'ambiente di esecuzione può rappresentare un dato rilevante per eseguire un'ulteriore ottimizzazione del codice.

- VES – Virtual Execution System: rappresenta l'equivalente della macchina virtuale Java per l'ambiente di Sun/Oracle. Il VES carica, realizza i collegamenti ed esegue i programmi scritti per il Common Language Runtime. Il VES adempie le sue funzioni di loader utilizzando le informazioni contenute nei metadati ed utilizza late binding per integrare moduli compilati separatamente, che possono essere anche scritti in linguaggi differenti. Il VES inoltre fornisce servizi durante l'esecuzione dei codici, che includono la gestione automatica della memoria, supporto per profiling e debugging, sandbox per la sicurezza analoghe a quelle Java e l'interoperabilità

con il codice non gestito come ad esempio componenti COM.



Come detto, quando viene compilato un programma .NET scritto in un qualsiasi linguaggio .NET (C#, VB.NET), il codice sorgente non viene direttamente tradotto in codice eseguibile binario ma in un codice intermedio, chiamato MSIL (Microsoft Intermediate Language), il quale viene poi interpretato dal CLR. Questo linguaggio intermedio è indipendente dall'hardware e dal sistema operativo. Solo in fase di esecuzione il CLR si occupa di tradurre il codice MSIL in codice eseguibile binario.

2.2 LE NOVITÀ DELLA VERSIONE 3.5

La prima versione del framework è stata rilasciata nel febbraio 2002, disponibile per windows 98, Me, NT 4.0, 2000, e XP. Da allora sono state rilasciate altre versioni co cadenza pressoché annuale. Con il rilascio del Framework 3.0 nel novembre 2006 sono stati introdotti quattro nuovi componenti per lo sviluppo:

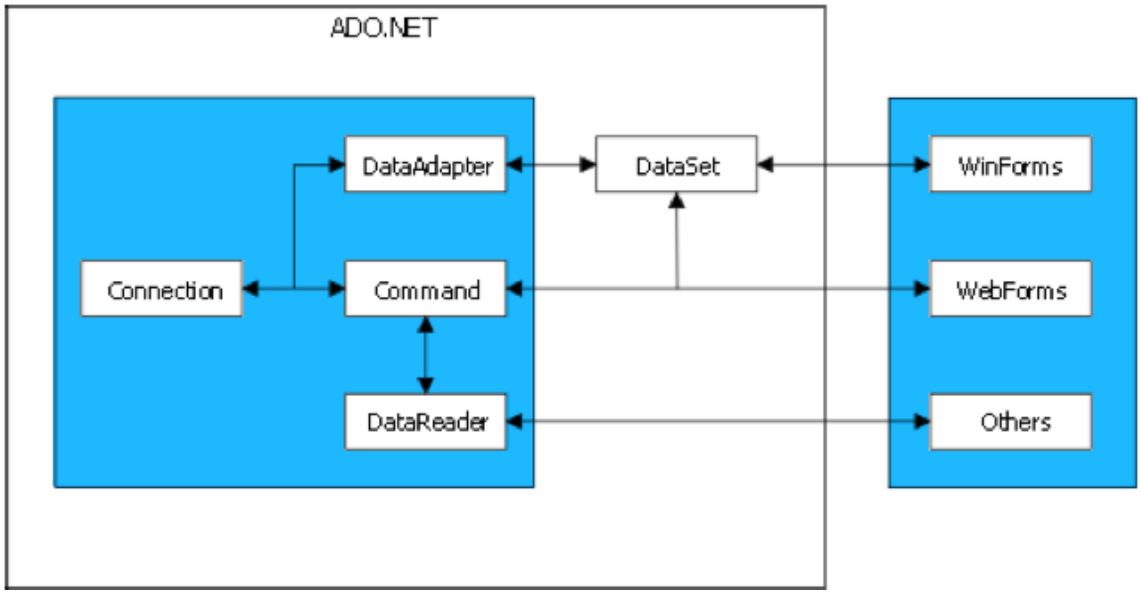
- Windows Presentation Foundation (WPF): rappresenta la libreria per la creazione delle interfacce grafiche, basata su grafica vettoriale e XML, riuscendo a sfruttare le potenzialità grafiche 3D dell'hardware e la tecnologia Direct3D.
- Windows Communication Foundation (WCF): un sottosistema per creare applicazioni distribuite con supporto alla logica transazionale.
- Windows Workflow Foundation (WF): per fornire alle applicazioni una tecnologia di progettazione, esecuzione e amministrazione di diagrammi di flusso.
- Windows CardSpace: per fornire alle applicazioni un metasistema d'identità dove gestire le password e i dati riservati in generale, presentando poi i dati aggregati in una sorta di carta delle identità virtuali.

Il 19 novembre 2007 è stata rilasciata la versione 3.5, utilizzata per questo progetto. Rispetto alla 3.0 include supporto al lambda calcolo e al metodo delle estensioni, tipi anonimi con inferenza statica, nuove funzionalità di rete, funzionalità AJAX ad ASP.NET e soprattutto Language Integrated Query (LINQ) che permette ai linguaggi .NET di effettuare queries di dati con sintassi simile a quella dell'SQL.

2.3 ADO.NET

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

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



Per una più facile interoperabilità, ADO.NET utilizza l'XML (eXtensible Markup Language) come formato nativo per i dati, mentre l'accesso a questi ultimi è stato progettato sulla base di un'architettura disconnessa. Ciò significa che le applicazioni rimangono connesse al database solamente per il tempo necessario per estrarre o aggiornare i dati: questo porta a numerosi vantaggi, primo su tutti il minore carico di lavoro subito

dal database, ed inoltre la maggior velocità di esecuzione del software, in quanto esso effettuerà le modifiche ai dati su una copia degli stessi tenuta in memoria, senza attendere il database server.

ADO.NET è costituito da due componenti fondamentali, il Data Provider e il DataSet.

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

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

possibile porre in relazione tra loro mediante oggetti DataRelation. È inoltre possibile applicare l'integrità dei dati nell'oggetto DataSet utilizzando gli oggetti UniqueConstraint e ForeignKeyConstraint.

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

2.4 ASP.NET

ASP.NET è un insieme di tecnologie di sviluppo di software per il Web commercializzate da Microsoft. Utilizzando queste tecnologie è possibile sviluppare applicazioni Web e Web service. Come tutte le applicazioni della famiglia Microsoft .NET, ASP.NET si basa sul CLR. Le applicazioni .NET sono significativamente più veloci e performanti rispetto a quelle realizzate utilizzando altre tecnologie di scripting, in quanto l'intero codice del sito Web è pre-compilato in pochi file dll.

ASP.NET si propone di semplificare la migrazione degli sviluppatori dalle applicazioni Windows alle applicazioni Web mettendoli in grado di generare pagine composte da tanti controlli widget simili a quelli usati dall'interfaccia utente di Windows. Con ASP.NET è quindi possibile sviluppare applicazioni Web utilizzando il paradigma dell'interfaccia grafica abbinata alla programmazione ad eventi, rendendone lo sviluppo del tutto simile a quello di una normale applicazione Windows.

Le classi della libreria Framework .NET si propongono, inoltre, di combinarsi ed interagire con le tecnologie esistenti, come ad esempio Javascript, in modo da attribuire un carattere di persistenza ad oggetti software, anche nell'ambito di un ambiente come il Web, che è intrinsecamente privo di stato.

2.4.1 Sviluppo in ASP.NET

I componenti principali di un'applicazione Web in ASP.NET sono le Web form. Ogni Web form rappresenta una pagina web ed è descritta da un file .aspx. Questo file contiene codice (X)HTML per la definizione del layout della pagina, che permette di inserire in essa, controlli web lato server e codice lato server.

È però sconsigliato inserire codice all'interno del file .aspx, Microsoft infatti suggerisce di utilizzare il modello code-behind. Esso consiste nello scrivere il codice, in risposta al verificarsi di certi eventi, in un file separato da quello .aspx, solitamente con estensione .aspx.cs o .aspc.vb in base al linguaggio utilizzato. Questo approccio permette di separare il design delle pagine dalla parte di codice vero e proprio che ne descrive il comportamento.

Per favorire il riutilizzo del codice, ASP.NET permette la creazione di controlli utente, dei blocchi di pagine che vengono registrati e utilizzati come controlli ASP.NET. Questi vengono creati come file .ascx che contengono codice (X)HTML come le pagine aspx, ai quali si affianca un file che, secondo il modello code-behind, ne definisce il comportamento.

I programmatori possono inoltre creare controlli personalizzati per le applicazioni ASP.NET. Questi controlli, a differenza dei controlli utente, non hanno un file ASCX, ma hanno tutto il codice compilato in un file DLL.

Le pagine ASP.NET sono accessibili tramite il protocollo HTTP che è intrinsecamente privo di stato. Spetta quindi ad ASP.NET il compito di conservare lo stato dell'applicazione.A questo scopo sono stati predisposti tre modi per conservare le informazioni relative allo stato dell'applicazione:

- **Stato sessione:** lo stato di sessione è formato da una serie di variabili definite dall'utente, mantenute lato server ed associate univocamente ad una sessione utente. Lato client, la sessione utente viene mantenuta attraverso i cookie, oppure codificando l'id di sessione nell'URL della richiesta.
- **View state:** si riferisce al meccanismo di mantenimento dello stato a livello pagina utilizzato dalle web form ASP.NET. Attraverso il view state vengono mantenute le informazioni relative allo stato dei web control e dei widget. Lo stato dei controlli è codificato è inviato al server ad ogni richiesta di pagina. Lo scopo principale del view state è di preservare le informazioni dei form attraverso il postback cioè quel procedimento che avviene quando il client invia le informazioni contenute in un form al server per farle elaborare ed ottenere in risposta un'altra pagina. Gli sviluppatori devono tener conto del fatto che di default le informazioni salvate nel viewstate (e quindi inviate ad ogni postback) non sono crittografate e che quindi non è difficile per un malintenzionato ricavarne dati sensibili o privati.
- **Server-side caching:** ASP.NET offre un oggetto “Cache” condiviso per tutta l'applicazione che può essere utilizzato per memorizzare diversi oggetti per una specifica quantità di tempo.

Con la versione 3.5 del Framework .NET è stata rilasciata anche la versione 3.5 di ASP.NET. Rispetto alle precedenti sono stati introdotti:

- Due nuovi controlli per la visualizzazione dei dati: List View e Data Pager.;
- ASP.NET AJAX;
- LINQ e tutte le altre novità della piattaforma .NET 3.5;

3 HTTPS e AUTENTICAZIONE SSL/TLS

Visto lo scarso livello di sicurezza fornito dalla tecnologia HTTP che invia e riceve tutti i dati in chiaro, si è deciso di utilizzare il protocollo sicuro HTTPS. Il protocollo HTTPS integra l'interazione di HTTP attraverso un meccanismo di crittografia di tipo SSL/TLS.

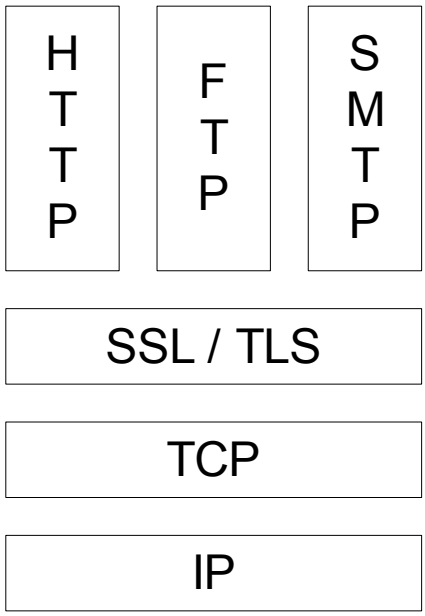
SSL (Secure Socket Layer) è un protocollo progettato dalla Netscape Communication Corporation per realizzare comunicazioni cifrate su internet. È supportato praticamente da tutti i browser Web ed è diventato lo standard per le comunicazioni sicure tra utenti e siti internet. Nel 1996 è stata rilasciata la versione 3.0 che è stata utilizzata come base di sviluppo per il successore Transport Layer Security (TLS) . SSL/TLS utilizza la crittografia per instaurare una connessione sicura fra due socket con le seguenti caratteristiche:

1. Negoziazione dei parametri di connessione fra client e server
2. Autenticazione: utilizza la crittografia asimmetrica, ovvero a chiave pubblica, per garantire l'identità di server e client. È prevista la certificazione del server e opzionalmente del client.
3. Privatezza: la crittografia asimmetrica viene utilizzata per stabilire una chiave di sessione ed un'algoritmo di cifratura simmetrico da utilizzare per assicurare la segretezza della trasmissione dei dati.
4. Affidabilità: il livello di trasporto include un controllo dell'integrità del messaggio basato su un codice MAC (Message Authentication Code) che utilizza funzioni hash per verificare che i dati che transitano da client a server non siano stati alterati durante la trasmissione.

Quindi lo scopo di questa tecnologia è di fornire sistemi di crittografia per

comunicazioni affidabili e riservate sulla rete, sfruttabili in applicazioni quali, ad esempio, posta elettronica, siti Web e sistemi di autenticazione.

Nella pila dei protocolli SSL va ad inserirsi tra i protocolli di trasporto (TCP) e i protocolli applicativi (HTML, SMTP, FTP) .



SSL/TLS può essere utilizzato in combinazione a qualsiasi protocollo che utilizzi TCP come protocollo di trasporto ma, solitamente, viene utilizzato in combinazione ad HTTP formando HTTPS o a POP/SMTP per rendere sicuro l'invio/ricezione di posta elettronica. Nel caso di HTTPS viene utilizzato per rendere sicure pagine Web nelle quali la sicurezza è critica, come nel caso di applicazioni di e-commerce o di home-banking. La porta solitamente utilizzata per l'accesso ad un servizio https è la 443 a differenza di http che utilizza la porta 80.

SSL/TLS utilizza metodi di cifratura e certificati per verificare l'identità di server e client e consiste di queste fasi principali:

- Negoziazione dell'algoritmo da utilizzare

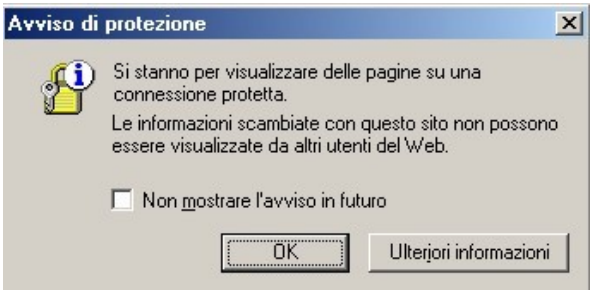
- Scambio di chiavi da utilizzare per cifrare la trasmissione e identificazione delle parti tramite l'utilizzo di certificati.
- Cifratura del traffico tra le parti utilizzando l'algoritmo e la chiave stabiliti nelle due fasi precedenti.Il protocollo SSL è composto da due sotto protocolli:
- Protocollo SSL Handshake, permette la reciproca autenticazione tra client e server e la negoziazione di un algoritmo di crittografia e delle relative chiavi prima che il livello applicazione trasmetta o riceva il primo byte. In questo modo SSL risulta indipendente dal protocollo di applicazione utilizzato, rendendosi completamente trasparente al livello superiore.
- Protocollo SSL Record, è interfacciato su un protocollo di trasporto affidabile come TCP. È utilizzato per l'incapsulamento dei dati provenienti dai protocolli superiori.

Le sessioni SSL sono suddivise in due fasi: la connessione e la fase applicativa. Durante la connessione, il client e il server si autenticano a vicenda e negoziano i parametri di sicurezza. Se il client accetta le credenziali del server, viene stabilita una chiave di cifratura per cifrare tutte le comunicazioni successive. Il sotto protocollo SSL Handshake si occupa dell'autenticazione tra le parti attraverso una serie di messaggi che inoltre permettono di selezionare un algoritmo di cifratura e un livello di sicurezza supportato da entrambe e di stabilire una chiave da utilizzare durante la fase successiva.

Durante la sessione applicativa il client e il server si scambiano informazioni tra loro in modo sicuro. Di questo si occupa il protocollo SSL Record. I dati pronti per essere inviati vengono suddivisi in frammenti e protetti dalle manomissioni attraverso un wrapper (un codice derivato dai

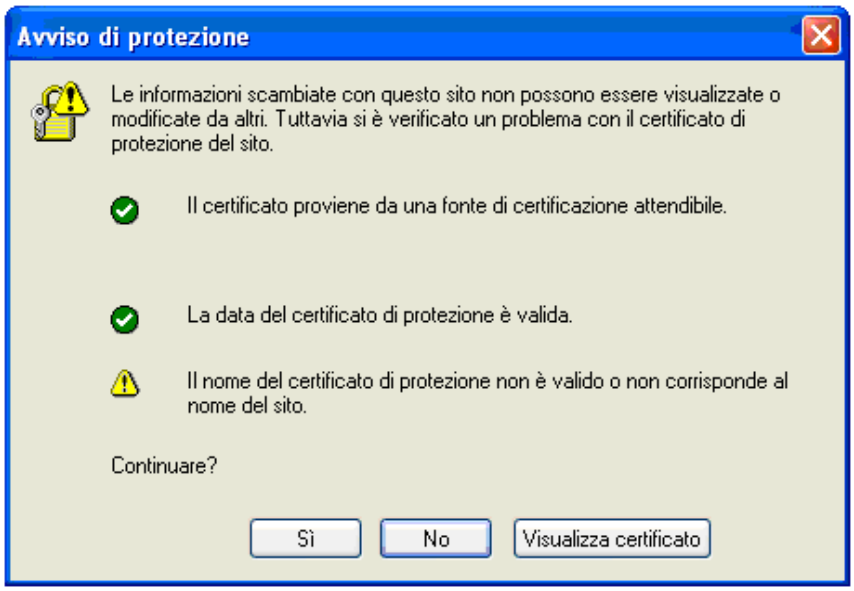
dati), dopodiché vengono cifrati e inviati assieme al wrapper.

L'indirizzo delle pagine visualizzate attraverso una connessione SSL è caratterizzato dal prefisso https. L'invio e la ricezione dei dati in connessione ad un server sicuro SSL viene segnalata dal browser con specifici messaggi di avvertimento e con un lucchetto chiuso in basso a destra (su Internet Explorer) o in basso a sinistra (su Mozilla Firefox).



Cliccando sul lucchetto si ha la possibilità di visualizzare i dati del certificato SSL del server (Autorità di Certificazione – date di rilascio e scadenza) .

Inoltre, se il certificato del server a cui si sta tentando di accedere non fosse considerato sicuro, il browser avvertirà l'utente del problema riscontrato con il certificato e permetterà di accettare comunque, di rifiutare o di visualizzarne i dati relativi.



Ciascuna delle due parti deve essere in grado di identificare l'altra. Infatti, il client deve essere certo che il server al quale si sta collegando non sia un falso server gestito da un malintenzionato con lo scopo di “rubare” dati personali; d'altra parte il server dovrebbe essere in grado di riconoscere il client per assicurarsi che esso sia autorizzato a visualizzare certe informazioni o effettuare certe operazioni. Per permettere ciò viene utilizzata la certificazione sia lato server che lato client.

Gli scopi del Protocollo SSL v3.0, in ordine di priorità, sono:

- Sicurezza della crittografia: SSL stabilisce un collegamento sicuro tra due sistemi.

- Interoperabilità: Programmatori di diverse organizzazioni dovrebbero essere in grado di sviluppare applicazioni utilizzando SSL 3.0, scambiandosi parametri della crittografia senza necessità di conoscere il codice l'uno dell'altro.

- Ampliamento: SSL cerca di fornire una struttura dentro la quale i nuovi metodi di crittografia possano essere incorporati se necessario. Questo fa sì che anche altri due aspetti importanti vengano soddisfatti: prevenire il bisogno di creare un nuovo protocollo ed evitare la necessità di implementare una nuova security library.

- Efficienza: Le operazioni di crittografia tendono a essere molto laboriose per la CPU, particolarmente le operazioni con le chiavi pubbliche. Per questa ragione l' SSL ha incorporato uno schema di session caching opzionale per ridurre il numero di collegamenti che hanno bisogno di essere stabiliti ex-novo. Particolare attenzione è stata posta nel ridurre l'attività sulla rete.

3.1 CRITTOGRAFIA

SSL/TLS utilizza la crittografia per verificare l'identità delle due parti e per trasmettere i dati in segretezza. Esistono due principali tipi di algoritmi crittografici:

- Algoritmi a chiave simmetrica.
- Algoritmi a chiave pubblica o asimmetrica.

Sono detti algoritmi a chiave simmetrica gli algoritmi che prevedono l'utilizzo della stessa chiave, sia per cifrare che per decifrare il messaggio. Quando si utilizza un algoritmo di questo tipo è necessario stabilire un modo per concordare la chiave condivisa senza che questa possa essere scoperta anche da un malintenzionato.

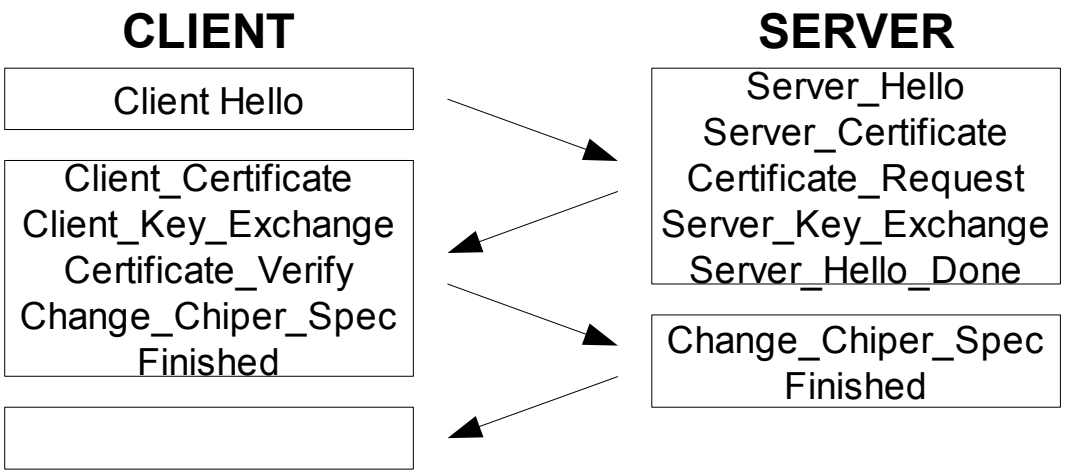
Gli algoritmi a chiave pubblica invece sono algoritmi di cifratura che prevedono due chiavi diverse, una per cifrare i dati, e una per decifrarli. In questo caso, la chiave usata per cifrare i dati può essere resa pubblica da chi vuole ricevere un messaggio perché comunque solo lui potrà decifrare i messaggi cifrati con tale chiave, utilizzando l'altra chiave che invece deve rimanere privata. Un problema degli algoritmi a chiave pubblica, riguarda la loro relativa pesantezza, che ne rende impraticabile l'utilizzo massiccio per cifrare grandi quantità di dati. Per questo, di solito, gli algoritmi di questo tipo vengono utilizzati per concordare una chiave condivisa da utilizzare con i più leggeri algoritmi a chiave simmetrica.

Gli algoritmi a chiave pubblica oltre a cifrare i dati, possono essere utili anche per verificare l'identità del mittente: infatti, se si mantiene privata la chiave per cifrare i dati e si pubblica quella necessaria a decifrarli, solo il vero mittente potrà scrivere un messaggio decifrabile con la chiave pubblica. In tal modo il destinatario, quando riesce a decifrare il messaggio

con la chiave pubblicata, può essere sicuro dell'identità del mittente posto che questa venga garantita da una adeguata certificazione o da conoscenza diretta. In particolare, di solito, vista la pesantezza di tali algoritmi, viene cifrata con la chiave pubblica solo un'impronta ottenuta con un algoritmo di hash dal messaggio originale; in questo modo, ricalcolando l'impronta e controllando se coincide, si è sicuri, oltre dell'identità del mittente, anche del fatto che il messaggio non sia stato modificato da un malintenzionato o a causa di errori di trasmissione.

3.2 PROTOCOLLO SSL HANDSHAKE

In questo capitolo verrà approfondito il sotto-protocollo SSL Handshake precedentemente solo accennato. Come già detto questo protocollo si occupa di autorizzare la comunicazione tra client e server.



Il protocollo SSL utilizza sia la crittografia a chiave pubblica che quella a chiave simmetrica. Una sessione SSL inizia sempre con uno scambio di messaggi chiamato SSL handshake. L'handshake permette al server e al client di autenticarsi a vicenda usando una tecnica a chiave pubblica, quindi di cooperare per la creazione delle chiavi simmetriche usate per la cifratura effettiva dei dati e il controllo delle intrusioni, in quanto la crittografia a chiave simmetrica risulta più veloce di quella a chiave pubblica. L'avvio di una connessione sicura può avvenire o da parte del client o da parte del server; nel caso sia il client ad iniziare, questo invierà un messaggio di client hello, dando inizio alla fase di Hello, e si porrà in attesa della risposta del server che avverrà con un server hello. Se è il server a voler iniziare la connessione, può farlo inviando una hello request alla quale il client risponderà iniziando la fase di Hello. A questo punto può

iniziare o meno uno scambio di certificati tra client e server. Un client abilitato può controllare che il certificato del server sia valido e che sia stato firmato da un'autorità di certificazione fidata (Certification Authority). Sia l'autenticazione del server che quella del client implica la cifratura di alcuni dati condivisi con una chiave pubblica o privata e la decifratura con la chiave corrispondente.

Nel caso in cui il client voglia verificare l'identità del server, può cifrare dei dati segreti con la chiave pubblica del server, in questo modo solo il vero server che possiede la corrispondente chiave privata può decifrare il dato. Se il server non riesce a decifrare i dati non potrà generare le chiavi simmetriche necessarie ad iniziare la fase di trasmissione di dati cifrati. Nel caso invece in cui il client voglia autenticarsi presso il server, può cifrare dei dati con la propria chiave privata, in modo che il server possa decifrarli utilizzando la chiave pubblica contenuta nel certificato del client ed essere così sicuro della sua identità. Se l'autenticazione è andata a buon fine, si può procedere, attraverso i messaggi di server key exchange e client key exchange, alla generazione delle chiavi per la cifratura e per l'autenticazione dei dati provenienti dal livello di applicazione. Successivamente, il server invierà un messaggio di hello done al client per segnalare la fine della fase di Hello, al quale seguiranno i messaggi di change chiper spec inviati da entrambi per verificare la correttezza dei dati ricevuti. Se tutto è avvenuto in maniera corretta, da questo punto in poi useranno gli algoritmi di sicurezza concordati. La fase di handshake terminerà con l'invio, da entrambe le parti, di un messaggio di finished che sarà il primo dato ad essere cifrato. Si è così creato un canale sicuro (tunnel SSL) tramite il quale client e server possono comunicare tra di loro in modo sicuro senza interferenze esterne.

3.3 CERTIFICATION AUTHORITY

Una Certification Authority è un'organizzazione di terza parte, pubblica o privata, abilitata a rilasciare certificati digitali che garantiscano l'associazione fra un particolare utente e una coppia di chiavi, rendendone disponibile la chiave pubblica. Nel certificato la CA dovrà inserire: i dati identificativi dell'utente, la sua chiave pubblica e la firma della CA stessa per assicurarne l'autenticità. Per poter fare questo, l'Authority deve essere in possesso di un certificato che la associa ad un coppia di chiavi rilasiciata da un'autorità superiore oppure dalla stessa (in quest'altro caso la CA si autocertifica).

Non è sempre necessario rivolgersi ad una CA per ottenere un certificato digitale, ma è possibile, con strumenti come OpenSSL, creare dei propri certificati. In questo caso l'utente funge egli stesso da Certification Authority, garantendo per se stesso. Spetta a chi riceve il certificato decidere se può essere sufficientemente fidato ed accettarlo oppure rifiutarlo.

Se tutte le persone interessate alla firma elettronica andassero dalla CA con un tipo diverso di certificato, l'operazione di gestire i diversi formati diventerebbe presto proibitiva. Per risolvere questo problema è stato sviluppato uno standard per i certificati chiamato X.509, arrivato attualmente alla versione V3.

3.4 CERTIFICATI X509

Lo scopo principale di X.509 è quello di descrivere e standardizzare la struttura dei certificati. Come introdotto nel capitolo precedente, un certificato deve contenere le informazioni utili ad identificare l'utente, la chiave pubblica a lui associata e la firma della CA.Nel sistema X.509, una Certification Authority rilascia un certificato che accoppia una chiave pubblica ad un Nome Distintivo (Distinguished Name) utilizzando lo standard X.500, oppure ad un Nome Alternativo (Alternative Name) come potrebbe essere un indirizzo e-mail o un record DNS.La struttura di un certificato X.509 V3 è la seguente:

Campo	**Significato**
Version	Numero della versione di X.509
Serial Number	Il certificato è univocamente identificato da questo numero più il nome della CA
Signature Algorithm	Algoritmo usato per firmare il certificato
Issuer	Nome X.500 della CA
Validity Period	Inizio e fine del periodo di validità
Subject name	L'entità proprietaria della chiave da certificare
Public key	La chiave pubblica del soggetto e l'ID dell'algoritmo che la usa
Issuer ID	Facoltativo: identificativo univoco di chi emette il certificato
Subject ID	Facoltativo: identificativo univoco del soggetto del certificato
Extensions	Sono state definite molte estensioni
Signature	La firma del certificato (firmata dalla chiave privata della CA)

3.5 GESTIONE DI CERTIFICATI CON OPENSSL

OpenSSL è un implementazione open source dei protocolli SSL e TLS. Lo strumento permette di gestire tutti gli aspetti relativi alla gestione dei certificati tramite un'interfaccia a linea di comando. È importante notare che, essendo le chiavi generate con questo procedimento non riconosciute da nessuna CA, ma firmate dal creatore stesso, molti browser segnaleranno la cosa all'utente per avvertirlo che i certificati potrebbero non essere affidabili.

Di seguito verrà descritta la procedura da eseguire per creare dei propri certificati senza ricorrere ad una CA.

Come prima cosa è necessario generare le chiavi da utilizzare per generare e firmare al posto della CA tutte le chiavi di cui si avrà bisogno. Il comando da digitare è il seguente:

openssl genrsa -des3 -out ca.key 1024

dove -des3 rappresenta l'algoritmo di cifratura utilizzato. Con tale comando verrà generata una chiave a 1024 bit che verrà salvata all'interno del file ca.key. Verrà inoltre richiesta una password per proteggere la chiave privata generata che sarà necessario inserire ogniqualvolta si utilizzerà la chiave per cifrare o firmare.

Il passo successivo consiste nel generare il certificato della propria “CA” sfruttando la chiave generata in precedenza per la firma. Il comando per effettuare l'operazione è:

openssl req -new -X509 -days 1000 -key ca.key -out ca.cer

la quale creerà nel file ca.cer un nuovo certificato X.509 con validità 1000 giorni (modificabile a piacere) sfruttando per la firma la chiave ca.key.

Ora si dovranno creare le chiavi che verranno utilizzate da server e client:

openssl genrsa -des3 -out server.key 1024

Successivamente è necessario creare una richiesta di certificato al certification authority (certification signed request):

openssl req -new -key server.key -out server.csr

Infine, la richiesta sarà inviata in modo sicuro alla certification authority in modo che la possa firmare:

openssl ca -cert ca.cer -in server.csr -keyfile ca.key -days 360 -out server.cer

Per ottenere il certificato in un formato distribuibile è ora necessario eseguire il seguente comando:

openssl pkcs12 -export -inkey server.key -in server.cer -out server.p12 -name "Server certificate"

Verra richiesta una password da utilizzare per la conversione. Come risultato si otterrà in formato PKCS12, il certificato e la relativa chiave pubblica firmati dalla CA.

Per evitare che alcuni software segnalino i futuri certificati da noi generati come non affidabili è sufficiente aggiungere tale certificato PKCS12 alla lista delle Trusted Root Store, che si trova nella sezione strumenti di amministrazione dentro al pannello di controllo.

Sarà ora possibile, tramite la consolle di gestione di IIS, configurare il Web server per rendere accessibile un particolare sito/applicazione Web attraverso il protocollo https, utilizzando il certificato precedentemente creato.

Potrebbe, per diversi motivi quali ad esempio la compromissione della chiave primaria, essere necessario invalidare un certificato nonostante questo sia ancora formalmente valido. È possibile gestire queste situazioni

tramite la certificate revocation list (CRL). Innanzitutto bisogna creare la propria CRL tramite il comando:

openssl ca -gencrl -keyfile ca.key -cert ca.cer -out mycrl.pem

il file mycrl.pem conterrà le informazioni dei certificati che sono stati revocati, anche la CRL, come i certificati creati, va inserita alla lista delle trusted root store. Per inserire un certificato nella CRL, e quindi revocarlo, si può utilizzare il seguente comando:

openssl ca -revoke client.cer -keyfile ca.key -cert ca.cer

In questo modo, il certificato contenuto nel file client.cer non potrà più essere utilizzato perché compare nella CRL.

4 CONNESSIONE WIRELESS

In informatica il termine Wireless (dall'inglese senza fili) indica un sistema di comunicazione tra dispositivi elettronici, che non fanno uso di cavi o portanti fisici. Per contro i sistemi tradizionali, basati su connessioni cablate, sono detti wired.Generalmente i dispositivi Wireless utilizzano onde radio a bassa potenza; tuttavia la definizione si estende anche ai dispositivi, meno diffusi, che sfruttano la radiazione infrarossa o il laser.

Un tempo, a causa del prezzo elevato degli apparecchi Wireless, questa tecnologia veniva utilizzata soltanto in caso di condizioni in cui l'uso di cavi era difficile o impossibile. Man mano che i prezzi diminuiscono, però, le WLAN (Wireless LAN) stanno entrando anche nelle case, permettendo la condivisione di dati e della connessione internet tra i computer della famiglia.Le reti locali Wireless possono utilizzare come mezzo trasmissivo le onde radio, la luce infrarossa o i sistemi laser. Le onde radio vengono utilizzate dalle reti di tipo Wi-Fi cioè reti che devono coprire ambienti eterogenei dove le diverse postazioni da collegare non sono necessariamente visibili.

Le reti basate su infrarossi vengono utilizzate per collegare dispositivi visibili direttamente, sono lente e spesso utilizzano dispositivi dedicati. Ad oggi sono praticamente in disuso, sostituite dalle reti Bluetooth.

Le reti basate su laser vengono utilizzate normalmente per collegare sotto-reti costruite utilizzando tecnologie diverse. Il laser viene utilizzato per la sua elevata velocità di trasmissione. Un esempio tipico è il collegamento delle reti di due edifici vicini. Il laser ha lo svantaggio di essere sensibile alle condizioni esterne e alla vibrazioni; infatti, anche queste tipologie di dispositivi sono considerate in disuso.

4.1 TECNOLOGIA WI-FI

Wi-Fi, abbreviazione di Wireless Fidelity, è un termine che indica dispositivi che possono collegarsi a reti locali senza fili (WLAN) basate sulle specifiche dello standard IEEE 802.11. Un dispositivo, anche se conforme alle specifiche, non può utilizzare il logo ufficiale Wi-Fi se non ha superato le procedure di certificazione stabilite dal consorzio Wi-Fi Alliance (Wireless Ethernet Compatibility Alliance), che collauda e certifica la compatibilità dei componenti wireless con gli standard 802. La presenza del marchio Wi-Fi su un dispositivo dovrebbe quindi garantirne l'interoperabilità con gli altri dispositivi certificati, anche se prodotti da aziende differenti.

Le reti Wi-Fi sono infrastrutture relativamente economiche e di veloce attivazione e permettono di realizzare sistemi flessibili per la trasmissione di dati usando frequenze radio, estendendo o collegando reti esistenti oppure creandone di nuove.

Con la tecnologia Wi-Fi si ha la possibilità di scambiare dati ad una velocità che può variare da 11 Mbit/s (standard 802.11b) a 54Mbit (standard 802.11g) fino a 100Mbit/s (standard 802.11n), il tutto senza utilizzare alcun cavo di rete, mantenendo quindi una completa libertà di movimento e risparmiando sui costi di cablaggio. Le tecnologie sfruttate nella

realizzazione di reti wireless sono sostanzialmente due: Narrowband e Spread Spectrum. La prima è meno diffusa e vincolata dal fatto che gli utenti sfruttano frequenze radio ben definite, limitando la banda entro un range molto ridotto. Queste frequenze necessitano di una opportuna licenza associata all'assegnazione di una particolare frequenza. La seconda, invece, è più diffusa nella realizzazione di WLAN. Questa tecnologia utilizza una banda maggiore e questo si traduce automaticamente in un segnale più forte.

La maggior parte delle reti WI-FI non prevede alcuna protezione da un uso non autorizzato. Questo è dovuto al fatto che all'atto dell'acquisto le impostazioni predefinite non impongono all'utente l'utilizzo di nessun metodo di protezione.I metodi per evitare utilizzi non autorizzati sono nati di pari passo con lo sviluppo di nuove tecnologie e la "rottura" di algoritmi precedenti. Il primo sistema sviluppato è stato il WEP, Wired Equivalent Privacy, che però soffre di problemi intrinsechi che lo rendono, di fatto, inutile. È possibile sopprimere la trasmissione dell'SSID di identificazione oppure limitare l'accesso a indirizzi MAC ben definiti, però si tratta di metodi facilmente aggirabili. Per sopperire ai problemi del WEP sono stati inventati i protocolli WPA e WPA2 che offrono livelli di sicurezza maggiori. Rispetto a WEP, il WPA aumenta la dimensione della chiave, il numero delle chiavi in uso, include un sistema per verificare l'autenticità dei messaggi migliore e quindi incrementa la sicurezza della WLAN, rendendola effettivamente analoga a quella di una rete su cavo. È stato dimostrato che è possibile forzare una connessione WPA in soli 60 secondi, perciò è stato definito lo standard WPA2 che come algoritmo di cifratura utilizza AES ed è attualmente considerato immune da ogni tipo di attacco.

4.2 TECNOLOGIA WAP

Il WAP (Wireless Application Protocol) è un protocollo di connessione per dispositivi senza fili.

La prima standardizzazione del protocollo è avvenuta con la versione WAP 1.0, nata nell'aprile del 1998. Gli obiettivi che il WAP forum si poneva erano quattro:

- Trasportare le informazioni e i servizi residenti in internet su rete cellulare digitale e/o altre reti wireless.
- Creare specifiche di protocollo globali che permettano la loro adozione su qualsiasi rete wireless
- Rendere disponibile la creazione di informazioni e applicazioni che siano fruibili da ogni tipo di rete wireless
- Utilizzare fin dove possibile le tecnologie di accesso all'informazione esistenti.

Questi obiettivi però andavano perseguiti tenendo conto delle limitazioni dovute alle ridotte capacità dei terminali e delle reti wireless.

Essendo WAP uno standard aperto, è imperativo che dispositivi, servizi e

applicazioni posseggano almeno questi 4 requisiti:

1. Interoperabilità: i terminali di differenti costruttori devono essere compatibili con qualsiasi servizio in rete.

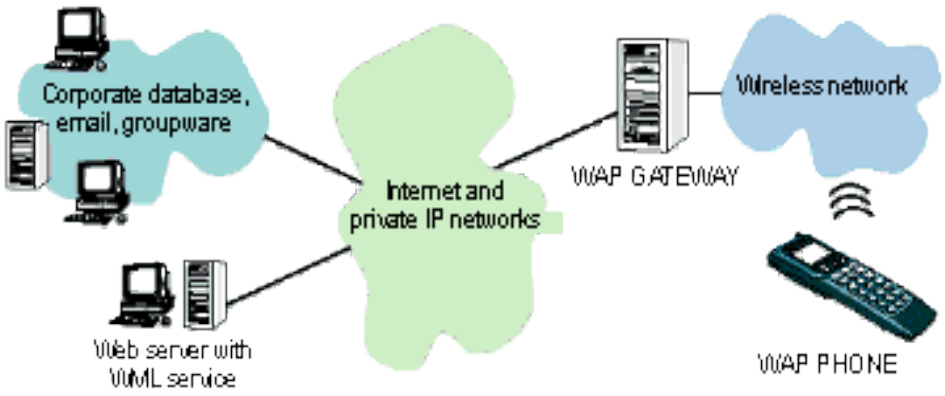
2. Scalabilità: gli operatori di rete devono poter integrare servizi e applicazioni con la massima flessibilità.

3. Efficienza: la qualità del servizio erogato deve essere adeguata alla rete di supporto.

4. Sicurezza: laddove richiesto si deve garantire l'integrità e la protezione dei dati sensibili catturabili da parti di terzi.

4.2.1 Protocollo WAP

Il protocollo WAP ha l'obiettivo di replicare, laddove possibile, quanto esistente nell'architettura WWW con alcune restrizioni imposte dalle intrinseche caratteristiche dell'ambiente wireless e dalle limitate risorse presenti nel client.

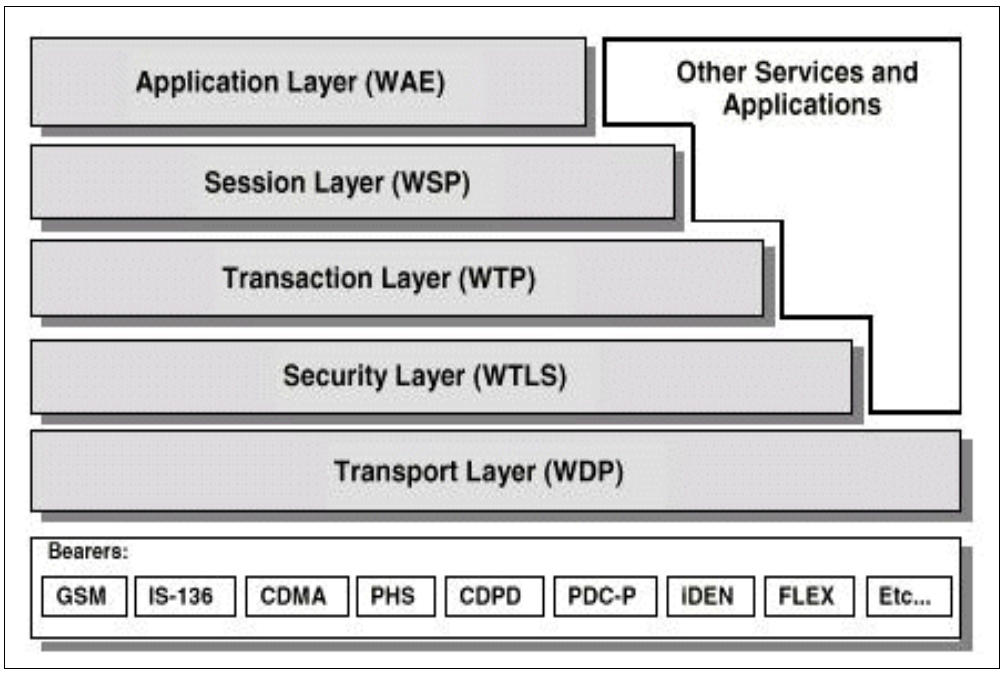


L'architettura WAP interpone tra client (anche chiamato terminale wireless) e origin server un gateway cui sono demandati tre compiti di fondamentale importanza:

1. Traslare in senso bidirezionale le istanze dall'ambiente WAP all'ambiente Internet utilizzando un protocol stack (WSP, WTP, WDP).
2. Codificare in un formato binario compatto la risposta del server allo scopo di ridurre lo spazio occupato dal dato nella rete wireless e viceversa.
3. Compilare eventuali script residenti nelle pagine WML da inviare a destinazione.

L'architettura della piattaforma WAP è organizzata in cinque livelli che e prevede un ambiente complessivo scalabile ed estensibile dedicato a sviluppi applicativi di tipo mobile communication.

Ciascun livello è accessibile dal livello superiore cosi come da altri servizi e applicazioni in senso trasversale.



La pila WAP può essere vista come composta da tre ambienti: l'ambiente applicativo (WAE), l'ambiente trasporto (WSP + WTP + WDP) e l'ambiente (opzionale) sicurezza (WTLS).

WAE (Wireless Application Environment) è un ambiente general-purpose sul quale è possibile sviluppare servizi e applicazioni per una grande varietà di dispositivi wireless. Nelle prime versioni era supportato solamente WML, un linguaggio basato su XML ottimizzato per l'utilizzo in dispositivi mobile. Dalle ultime versioni invece, ci si è posti l'obbiettivo di

far convergere Wap e Web e per questo il supporto è stato esteso anche al linguaggio XHTML.

WSP (Wireless Session Protocol) interfaccia direttamente il livello applicativo (WAE) ricoprendo lo stesso ruolo di HTTP nell'architettura Internet; i servizi offerti al superiore livello Application (WAE) sono orientati allo scambio dei contenuti in senso bidirezionale client/server

WTP (Wireless Transaction Protocol) gestisce l'invio/ricezione di pacchetti provenienti dagli strati superiori, occupandosi degli ack e dell'eventuale ritrasmissione di pacchetti errati o non giunti a destinazione.

WDP (Wireless Datagram Protocol) si interpone tra lo strato fisico (bearer) e il WTP, con lo scopo di offrire a quest'ultimo una serie di servizi di trasporto dei pacchetti in modo indipendente dal bearer su cui poggia.

WTLS si trova opzionalmente tra lo strato WDP e lo strato WTP e si occupa della sicurezza delle comunicazioni. Questo protocollo deriva dall'equivalente web TLS, incorporando, rispetto a questo, alcune ottimizzazioni per funzionare meglio su reti a banda ristretta.

5 INTERFACCIA WEB GESTIONALE PER STUDI LEGALI

Come già detto, in precedenza, all'interno dell'azienda ospitante, è stato sviluppato un gestionale per singoli avvocati e per studi legali di medie e grandi dimensioni consistente in un'applicazione desktop. L'applicazione Web realizzata durante il tirocinio è un'interfaccia Web che deve fornire alcune delle funzionalità offerte dall'applicazione desktop in modo da renderle accessibili da qualsiasi dispositivo in qualsiasi luogo purché connessi ad internet. In particolare la demo sviluppata permette la visualizzazione dello storico clienti, l'aggiunta, la rimozione e la modifica di un cliente dello storico, la visualizzazione dello storico delle pratiche, e la visualizzazione, l'inserimento e la modifica degli eventi presenti in agenda.

5.1 DESCRIZIONE ESECUTIVA

Prima di passare alla realizzazione dell'applicativo non è stato necessario progettare il database in quanto la web-application deve utilizzare lo stesso database utilizzato dall'applicativo desktop.

Per quanto riguarda l'interfaccia grafica, l'unico requisito è che questa sia più simile possibile a quella dell'originale applicazione desktop, ovviamente con i dovuti adattamenti necessari per renderla fruibile anche su schermi di dimensioni ridotte.

5.1.1 Modello MVC

Prima di procedere alla stesura del codice è stato necessario progettare l'architettura dell'applicazione. Descrivere l'architettura informatica di una realtà significa elencarne le sottoparti costituenti ed illustrare i rapporti che le legano l'una all'altra. L'architettura considera gli aspetti che sono inerenti alla comunicazione tra le parti, si focalizza sulle modalità di interazione, tralasciando i dettagli di funzionamento interni. Nei moderni sistemi software (per esempio nelle applicazioni orientate agli oggetti) le parti interagiscono tra loro per mezzo di interfacce che suddividono in modo netto ciò che non è direttamente accessibile dall'esterno da ciò che è pubblico. L'architettura si concentra unicamente sul secondo aspetto dei due, tralasciando i dettagli interni che in generale non influenzano il modo con cui i componenti si relazionano tra loro.

Tra gli stili architetturali quello che meglio si presta per la progettazione di applicazioni in ambiente .NET è l'architettura a livelli: questo stile prevede la strutturazione logica di un sistema software in strati sovrapposti (layer) tra loro comunicanti, ciascuno caratterizzato da una forte omogeneità funzionale. La forma più nota e usata riguarda l'architettura a tre livelli composta da:

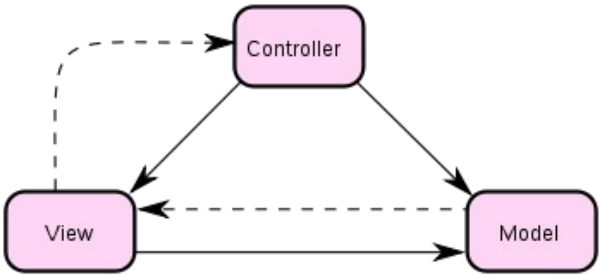
1. **interfaccia utente** o strato di presentazione. È lo strato di presentazione dell'applicazione; a questo livello viene definita la rappresentazione dei dati e l'interfaccia utente utilizzando pagine e controlli Web. I dati rappresentati sono forniti dai livelli inferiori;

2. **livello logico** o strato di business. È il livello di descrizione delle

entità logiche che descrivono i processi di business utilizzati all'interno dell'applicazione. Tali entità sono delle classi vere e proprie, con attributi e metodi utili a rappresentare la logica applicativa dell'applicazione. All'interno di questo strato vengono definite tecniche di gestione dei dati come, ad esempio, il loro inserimento nella cache della pagina o eventuali codifiche o decodifiche. È quindi questo il livello in diretta collaborazione con l'interfaccia utente;

3. **livello di accesso ai dati.** È il livello d'interazione con la base di dati legata al sito, dove si compie fisicamente la connessione e si eseguono query di selezione, inserimento, aggiornamento o cancellazione. È in questo strato che si determina l'efficienza della conservazione e della consultazione dei dati.

Tra le architetture a livelli il modello più adatto allo sviluppo di applicazioni Web utilizzando ASP.NET è il Model-View-Controller (MVC), dove il Model è rappresentato dal livello dati indipendente dall'interfaccia grafica, il View l'interfaccia grafica composta dagli elementi visuali come bottoni e altri controlli e il Controller è l'interfaccia tra dati, interfaccia grafica e input.



Grazie all'ASP.NET MVC Framework, sviluppato da Microsoft come aggiunta ad ASP.NET, è possibile sviluppare applicazioni in ASP.NET seguendo il modello MVC, in alternativa al Web Forms solitamente utilizzato per la creazione di questo tipo di applicazioni.

La scelta di utilizzare MVC, permettere di separare il lavoro di sviluppo vero e proprio da quello di design, riducendo la complessità dell'architettura e incrementando la flessibilità e la manutenibilità del codice.

5.2 PROBLEMATICHE RISCONTRATE

Di seguito verranno illustrati i principali problemi incontrati durante la realizzazione del progetto, con le relative soluzioni.

5.2.1 Sicurezza e Login

Il primo problema incontrato nella realizzazione del progetto è stato quello relativo alla sicurezza. Infatti, vista la presenza di dati sensibili riguardanti lo studio e i clienti, era richiesto un buon livello di segretezza durante il transito dei dati nella rete internet, e soprattutto di limitare l'accesso solo agli utenti autorizzati. Inoltre alcune funzionalità nell'agenda dovevano essere accessibili solo ad un certo gruppo di utenti.

Prima di tutto si è deciso di cifrare le comunicazioni utilizzando la tecnologia HTTPS/SSL già vista nel capitolo 3 per evitare che i dati transitino in chiaro attraverso la rete internet. È stato necessario creare un certificato auto-firmato utilizzando OpenSSL come illustrato nel paragrafo 3.4 e successivamente, configurare il server Web IIS, presente nel sistema operativo Windows XP Professional, per accettare connessioni HTTPS utilizzando il certificato appena creato. Come ultimo passo si è deciso di impedire l'accesso attraverso il protocollo in chiaro HTTP e fare in modo che la pagina di errore che si ottiene quando si tenta di accedere utilizzando tale protocollo reindirizzi direttamente alla corrispondente pagina HTTPS.

Per quanto riguarda l'accesso all'applicazione, che deve essere riservato ai soli utenti autorizzati, è stata predisposta una pagina di Login, dove l'utente dovrà inserire nome utente e password per poter accedere. Per evitare che un malintenzionato, una volta riuscito ad ottenere una copia

del Database possa recuperare tutte le password degli utenti, si è scelto di memorizzare nel database non la password stessa, bensì, il suo hash MD5 che dovrà essere confrontato con l'hash MD5 della password inserita dall'utente. Inoltre, per rendere meno efficaci attacchi come il brute force, si è deciso di permettere solo 3 tentativi di accesso falliti per ogni sessione e solo 5 tentativi falliti a distanza di massimo un minuto l'uno dall'altro provenienti dallo stesso IP, dopodiché tutte le richieste provenienti da quell'IP verranno bloccate per un certo periodo di tempo.

Per bloccare i tentativi falliti dopo il terzo nella stessa sessione è bastato mantenere in una variabile di sessione il contatore degli accessi falliti e disabilitare il form di login dopo il terzo tentativo. Per bloccare l'accesso dopo 5 tentativi falliti provenienti dallo stesso IP invece, è stata predisposta una classe Attempt che memorizza il timestamp dell'ultimo accesso e il numero di accessi consecutivi a distanza di meno di un minuto l'uno dall'altro. Si è poi allocato un dizionario che ad ogni indirizzo IP che tenta di collegarsi associa un oggetto della classe Attempt. Ad ogni tentativo di accesso si controlla se quell'IP ha già superato il numero massimo di tentativi e, se il tentativo di accesso fallisce, si aggiorna l'istanza di Attempt corrispondente. Di seguito verrà riportato il codice che si occupa della parte che si occupa della sicurezza nella pagina di login.

```
private class Attempt {
        private int NumAcc;
        private DateTime Last;
         public Attempt(String IP)        {        NumAcc = 0;        }
        public bool loginFailed() {  //ritorna false se quinto tentativo in 1 minuto, true altrimenti
                DateTime now = DateTime.Now;
                if (NumAcc == 0) {
                        NumAcc = 1;
                        Last = now;
                }       else {
                        if (now.Subtract(Last) > TimeSpan.FromMinutes(1))      NumAcc = 1;
                        else NumAcc++;
                        Last = now;
                }
                if (NumAcc >= numAccessiIP) return false;
                        return true;
        }
}
        protected void Page_Load(object sender, EventArgs e)
        {
                String IP = Request.UserHostAddress;
                if (AttemptDic.ContainsKey(IP) && AttemptDic[IP].getNumAcc() >= numAccessiIP)
                        if (DateTime.Now.Subtract(AttemptDic[IP].getLast()) > TimeSpan.FromMinutes(1))
                                AttemptDic.Remove(IP);
                        else        {
                                Response.Write("<br>" + numAccessiIP + " accessi falliti a distanza di un minuto.”
                                +  “Accesso temporaneamente bloccato");
                                form1.Visible = false;
        } }
         protected void BtnLogin_Click(object s, EventArgs e)   {
                string ConnectionString = Utility.Database.ConnectionString(Utility.Database.getDbName(Page));
                SqlConnection conn = new SqlConnection(ConnectionString);
                conn.Open();
                SqlCommand command = new SqlCommand("Select * from tblUtenti where username = @Usr", conn);
                command.Parameters.Add("@Usr", SqlDbType.VarChar).Value = TxtUser.Text;
                SqlDataReader reader = command.ExecuteReader();
                 if (!reader.HasRows)  {
```

```
                    Login_Error();
                    conn.Close();
                     return;
            }       else        {
                    reader.Read();
                     string hashedPwd = MakeSHA1(TxtPasswd.Text);
                    if (hashedPwd.CompareTo(reader.GetString(1)) == 0)    {
                            Response.Write("Ok... Login corretto");
                            Session["user"] = Anagrafica.getAnagrafica(TxtUser.Text,
                                    Utility.Database.getDbName(Page));
                            AttemptDic.Remove(Request.UserHostAddress);//rimuovo tentativi da questo IP
                            loginOk();
                    }       else        Login_Error();
            }       conn.Close();
}
protected void Login_Error()    {
        Response.Write("Username o password Errati");
        String IP = Request.UserHostAddress;
        if (!AttemptDic.ContainsKey(IP))  AttemptDic.Add(IP, new Attempt(IP));
        if (!AttemptDic[IP].loginFailed()) {
                form1.Visible = false;
                Response.Write("<br>" + numAccessiIP + " accessi falliti a distanza di un minuto. Accesso” +
                ”temporaneamente bloccato");
        }
        DateTime now = DateTime.Now;
        for (int i = AttemptDic.Count - 1; i > -1; i--)   {
                if (now.Subtract(AttemptDic.ElementAt(i).Value.getLast()) > TimeSpan.FromMinutes(1))
                 AttemptDic.Remove(AttemptDic.ElementAt(i).Key);
        }
        String SessionID = this.Session.SessionID;
        if (SessionDic.ContainsKey(SessionID))      {
                SessionDic[SessionID]++;
                if (SessionDic[SessionID].Equals(numAccessiSessione))           {
                        form1.Visible = false;
                        Response.Write("<br>" + numAccessiSessione + " accessi consecutivi falliti. Accesso “
                                + “bloccato");
                        SessionDic.Remove(SessionID);
                        Session["formDisabled"] = true;
        } }     else SessionDic.Add(SessionID, 1);  } }
```

5.2.2 Controlli utente personalizzati

Per rendere l'interfaccia più gradevole e facilitarne l'utilizzo anche da dispositivi mobili dotati di uno schermo di dimensioni ridotte si è reso indispensabile personalizzare alcuni dei controlli web standard presenti nella libreria .NET.

Dal controllo web GridView è stato creato un controllo personalizzato MyGridView che a differenza di quello originale:

- Evidenzia le righe al passaggio del mouse.
- Permette di selezionare una riga della grid cliccando in un qualsiasi punto della riga e non solo nell'apposito tasto “seleziona”.
- Aggiunge una richiesta di conferma prima dell'eliminazione di una riga.
- Permette, attraverso un menù che appare cliccando con il tasto destro sulla riga di intestazione o con il sinistro su un apposito tasto, di scegliere quali colonne visualizzare e quali nascondere.

Per aggiungere queste funzionalità al controllo GridView è stato modificato il metodo che gestisce l'evento di creazione di una riga,

aggiungendo ad ogni riga gli attributi javascript necessari:

```
protected override void OnRowDataBound(GridViewRowEventArgs e)          {
        base.OnRowDataBound(e);
        if (e.Row.RowType == DataControlRowType.Header)
                e.Row.Attributes["oncontextmenu"] = "return ePop" + this.ID +"();";
        if (e.Row.RowType != DataControlRowType.DataRow)   return;
        ImageButton lnkBtn;
         int firstDataCellIndex = 0;
         if (this.Columns.Count > 0 && this.Columns[0].GetType() == typeof(Utility.MyButtonField))          {
                lnkBtn = (ImageButton)e.Row.Cells[0].Controls[0];
                lnkBtn.ToolTip = "Seleziona";
        }
        GridView g = this;
        for (int i = firstDataCellIndex; i < e.Row.Cells.Count; i++)          {
                e.Row.Cells[i].Attributes["onclick"]    =    this.Page.ClientScript.GetPostBackClientHyperlink(this,
                        "Select$" + e.Row.RowIndex);
                e.Row.Cells[i].Attributes["onmouseover"] = "this.style.cursor='pointer';";
        }
        /*Recupero colori di sfondo e testo della riga*/
        string className;
        if (e.Row.RowState == DataControlRowState.Normal)   className = "normalGridRow";
        else          className = "alternateGridRow";
        //evidenzio la riga al passaggio del mouse
        e.Row.Attributes["onmouseover"] = "this.className='selectedGridRow'";
        e.Row.Attributes["onmouseout"] = "this.className='" + className + "';";
}
```

Dal controllo web Calendar è stato creato un controllo personalizzato EventCalendar che una volta impostato l'attributo DataSource con il riferimento ad una collection di oggetti (che possono essere di diverso tipo) che implementano l'interfaccia Event contrassegna i giorni in cui sono presenti eventi in agenda con un simbolo indicato dall'attributo imageUrl.

```
protected void EventCalendarDayRender(object sender, DayRenderEventArgs e)          {
        CalendarDay d = ((DayRenderEventArgs)e).Day;
        TableCell c = ((DayRenderEventArgs)e).Cell;
        // If there is nothing to bind
        if (this.DataSource == null)
                return;
        List<Event> dt = this.DataSource;
        c.CssClass = EmptyDayCssClass;
        foreach (Event dr in dt)            {
        if (!d.IsOtherMonth && Convert.ToDateTime(dr.getShowDate()).ToShortDateString()
                == d.Date.ToShortDateString())   {
                if (c.Controls.Count == 1)          {
                        System.Web.UI.WebControls.Image img = new System.Web.UI.WebControls.Image();
                        img.ImageUrl = ImageUrl;
                        img.CssClass = ImageCss;
                        img.ToolTip = dr.getTitle() + "\n";
                        c.Controls.Add(img);
                        c.CssClass = EventDayCssClass;
                }
                else {
                        ((Image)c.Controls[1]).ToolTip += dr.getTitle() + "\n";
}} }
```

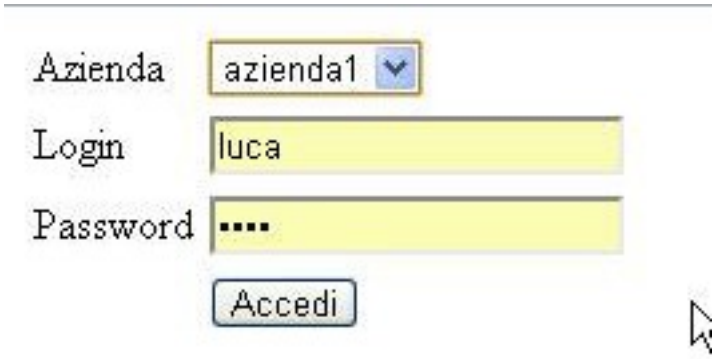
5.2.3 Pop-Up modali

In alcune parti dell'applicazione web è stata richiesta la realizzazione di pop-up modali, cioè di finestre che si aprono “sopra” la finestra principale al verificarsi di un evento e che richiedono di essere chiuse prima di poter tornare alla pagina principale. Di fatto quindi all'apertura del pop-up, tutti gli altri controlli presenti nella pagina principale devono essere disabilitati. Per disabilitare tutti i controlli indipendente da quanti e quali controlli sono presenti nella pagina, si è deciso di utilizzare un div (tag che nel linguaggio HTML definisce una sezione di una pagina e tramite il quale è possibile gestire posizione e stile di un gruppo di elementi), trasparente con z-index (indice che rappresenta il “piano” parallelo allo schermo su cui poggia il form) maggiore di quello della pagina principale ma appena minore rispetto a quello del div che rappresenta il pop-up. In questo modo il pop-up risulterà in primo piano, mentre la pagina principale verrà coperta dal div invisibile e i controlli in essa presente non saranno più utilizzabili. È stato quindi creato un controllo denominato “hidePanel” che rappresenta il pannello trasparente.

5.3 MANUALE UTENTE

5.3.1 Accesso

Per accedere all'applicazione è sufficiente aprire un browser web, e inserire l'URL dell'applicazione. Una volta connessi, viene presentata la pagina di login dove l'utente autorizzato può inserire il suo nome utente e la password e cliccare sul pulsante “Accedi”.



Come si può vedere dall'immagine, è possibile, nel caso nello stesso server siano presenti i database di più studi legali, scegliere a quale accedere.

5.3.2 Home page e menu laterale

Se l'autenticazione dell'utente è andata a buon fine, l'applicazione presenterà l'home page. Questa è composta da un menù laterale sulla destra e un pannello orizzontale in alto: essi saranno presenti in ogni pagina a parte quella di login. La parte restante, che è diversa per ogni pagina, nel caso dell'home page sarà vuota.

Nella barra laterale a destra sono presenti tre pulsanti:

- Clienti: permette di visualizzare la pagina contenente la lista clienti e dalla quale si possono aggiungere, rimuovere o modificare i clienti.

- Agenda: visualizza gli eventi in agenda, e permette di inserirne di nuovi, oltre a rimuovere o modificare quelli esistenti.
- Pratiche: permette di visualizzare lo storico delle pratiche dello studio legale.

Nella barra superiore sono presenti tre link:

- L'icona dell'applicazione e la scritta “Alia Memo” che puntano entrambi all'home page.
- Il pulsante Logout che permette di scollegarsi ed azzerare la sessione.

5.3.3 Lista clienti

Cliccando sul pulsante clienti della barra laterale, si apre la pagina “Lista clienti” che, inizialmente, mostrerà una lista di massimo 20 clienti. Se il numero di clienti è maggiore di 20 la tabella sarà suddivisa in più pagine, tra le quali sarà possibile navigare utilizzando i tasti numerici al di sotto della stessa.

Inserendo una parola qualsiasi nella casella di testo in alto a destra e cliccando sul tasto “cerca” alla sua destra è possibile cercare tra tutti i clienti quelli che contengono quella parola in uno dei campi visualizzabili sulla tabella. La stella verde, in alto a sinistra, permette di accedere alla pagina di creazione di un nuovo cliente da dove sarà possibile inserire tutti

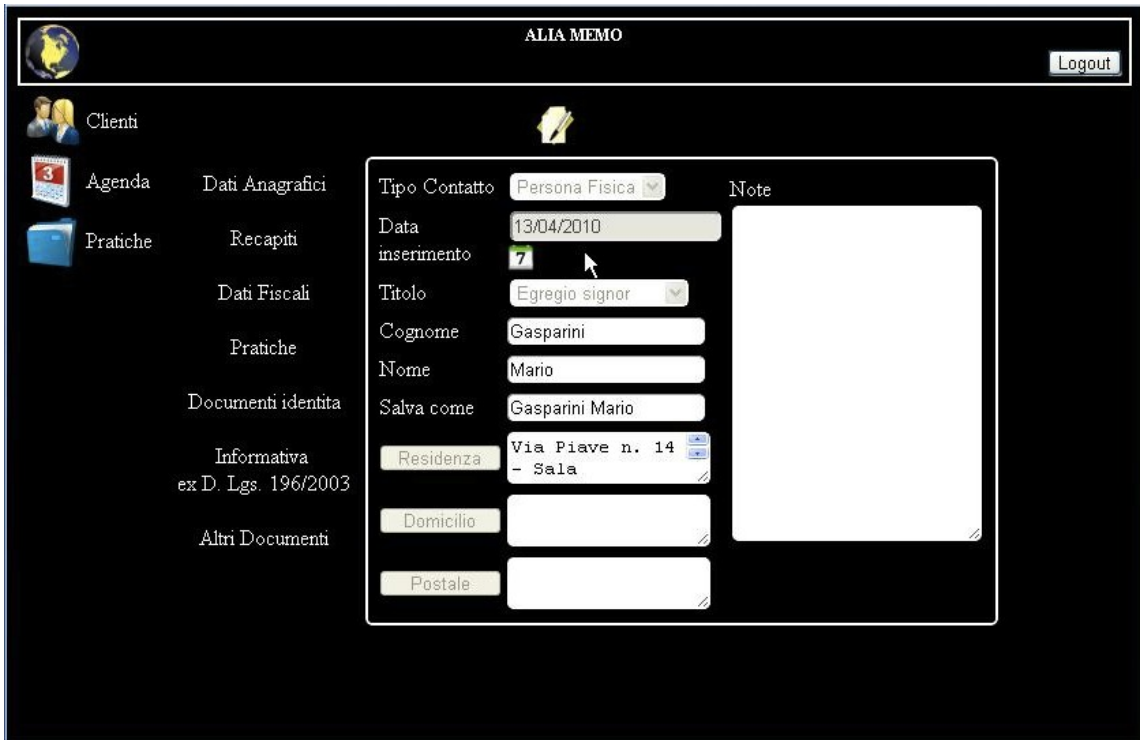
i dettagli ad esso relativi. All'inizio di ogni riga ci sono due pulsanti, il primo permette di selezionare un cliente e visualizzarne il dettaglio (che può essere fatto anche cliccando in un qualsiasi punto della riga), mentre il secondo, a forma di cestino, permette, dopo una conferma, di eliminare il cliente di quella riga.

Cliccando con il tasto sinistro su una qualsiasi intestazione di colonna è possibile ordinare i clienti in base al contenuto di quella colonna.

Cliccando invece con il tasto destro sulla riga di intestazione compare un pop-up che permette di scegliere quali colonne visualizzare. In dispositivi che non possiedono il tasto destro è possibile far comparire lo stesso menù cliccando sulla freccia blu verso il basso, presente sulla destra della riga di intestazione.

5.3.4 Dettaglio clienti

Come detto in precedenza, cliccando sul primo pulsante della riga nella lista clienti, o in un punto qualsiasi della riga, si apre il dettaglio del cliente selezionato:



Questa pagina si apre anche nel caso si prema il pulsante per la creazione di un nuovo cliente; in tal caso tutti i campi saranno vuoti.

Da questa pagina è possibile visualizzare tutti i dettagli relativi al cliente e, cliccando sull'opportuno pulsante presente sopra al form, attivare i controlli per permettere la modifica dei diversi campi. Per l'inserimento degli indirizzi è previsto un form a parte che apparirà dopo aver premuto il pulsante relativo e che faciliterà l'inserimento dell'indirizzo. Selezionando

una particolare provincia, infatti, si otterrà la lista di tutti i comuni con il Relativo CAP e, nel caso di città con più zone aventi CAP diversi, anche la lista delle diverse zone con i relativi CAP. Per confermare le modifiche è necessario cliccare sul pulsante “salva” (che comparirà dopo aver premuto il tasto “modifica”). Cliccando sul pulsante “salva ed esci” si confermeranno le modifiche e si tornerà subito sulla pagina “Lista clienti”. Cliccando sul pulsante “modifica” si farà anche comparire un tasto a forma di cestino che permette di eliminare il cliente di cui si stanno visualizzando i dettagli.

Di default viene visualizzato per primo il form contente i dati anagrafici. Cliccando invece sul link “recapiti” è possibile vedere il relativo form dove sono presenti campi adatti ad inserire diversi tipi di recapiti come mail e numero di telefono. Nel caso dei numeri di telefono, per inserire il prefisso internazionale sarà sufficiente selezionare la nazione corrispondente. Nel caso delle mail è presente un controllo di validità sul formato della mail.

Cliccando sul link “dati fiscali” sarà possibile visualizzare il relativo form dove sarà possibile visualizzare o inserire tutti i dati fiscali relativi al cliente. Se sono stati inseriti tutti i dati necessari al calcolo del codice fiscale, cliccando sull'apposito tasto, si può calcolare e inserire automaticamente il codice fiscale. Se il codice fiscale è già presente sarà effettuato un confronto con quello appena calcolato per verificarne la validità e, in caso siano diversi, verrà proposto di sostituire il codice fiscale presente con quello appena calcolato.

I link pratiche, documenti d'identità, informativa ex d. lgs. 196/2003 e altri documenti non sono attivi in quanto le funzionalità relative non sono previste nella demo sviluppata.

5.3.5 Agenda

Cliccando sul pulsante “Agenda” sulla barra laterale si passa alla visualizzazione dell'agenda. Tramite questa pagina è possibile visualizzare, inserire o modificare gli eventi presenti in agenda.

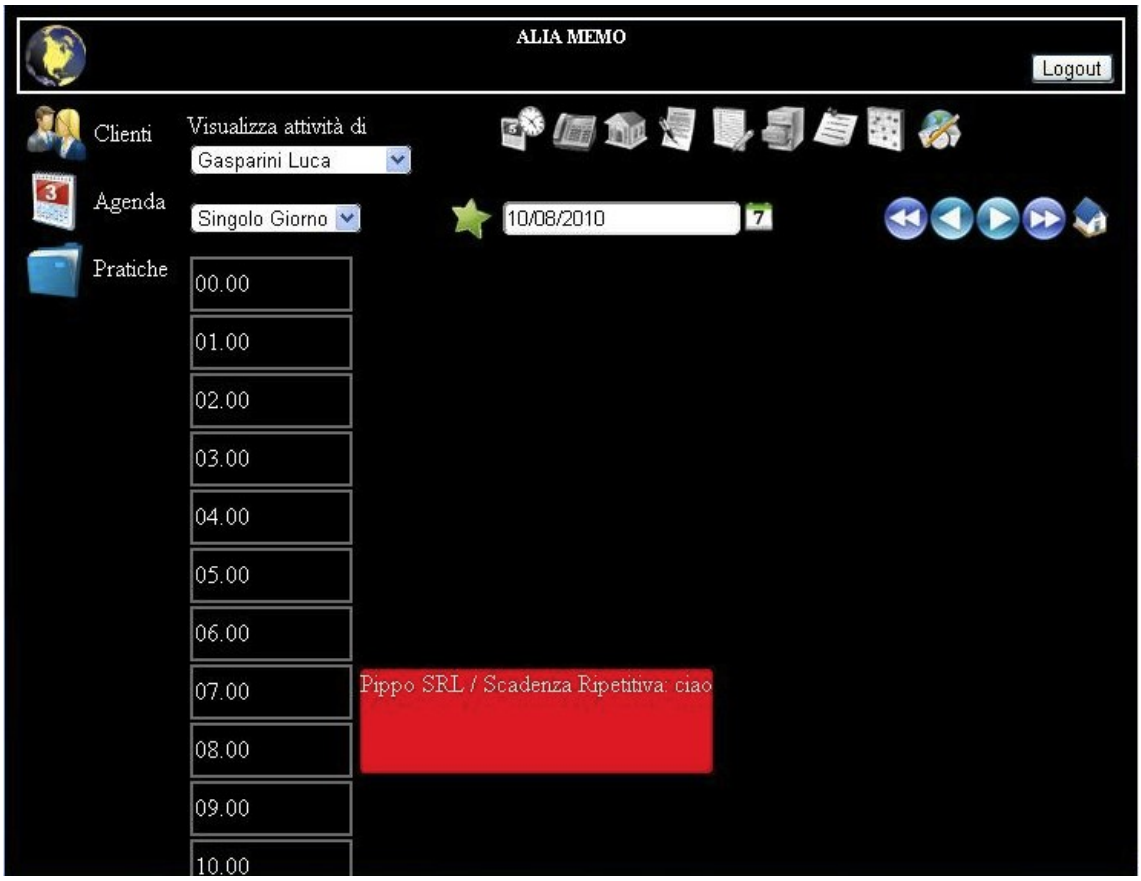
Gli eventi possono essere di diversi tipi:

- Adempimenti
- Annotazioni
- Appuntamenti
- Atti
- Cancellerie
- Scadenze ripetitive
- Udienze

La pagina di visualizzazione agenda può essere presentata in due modalità:

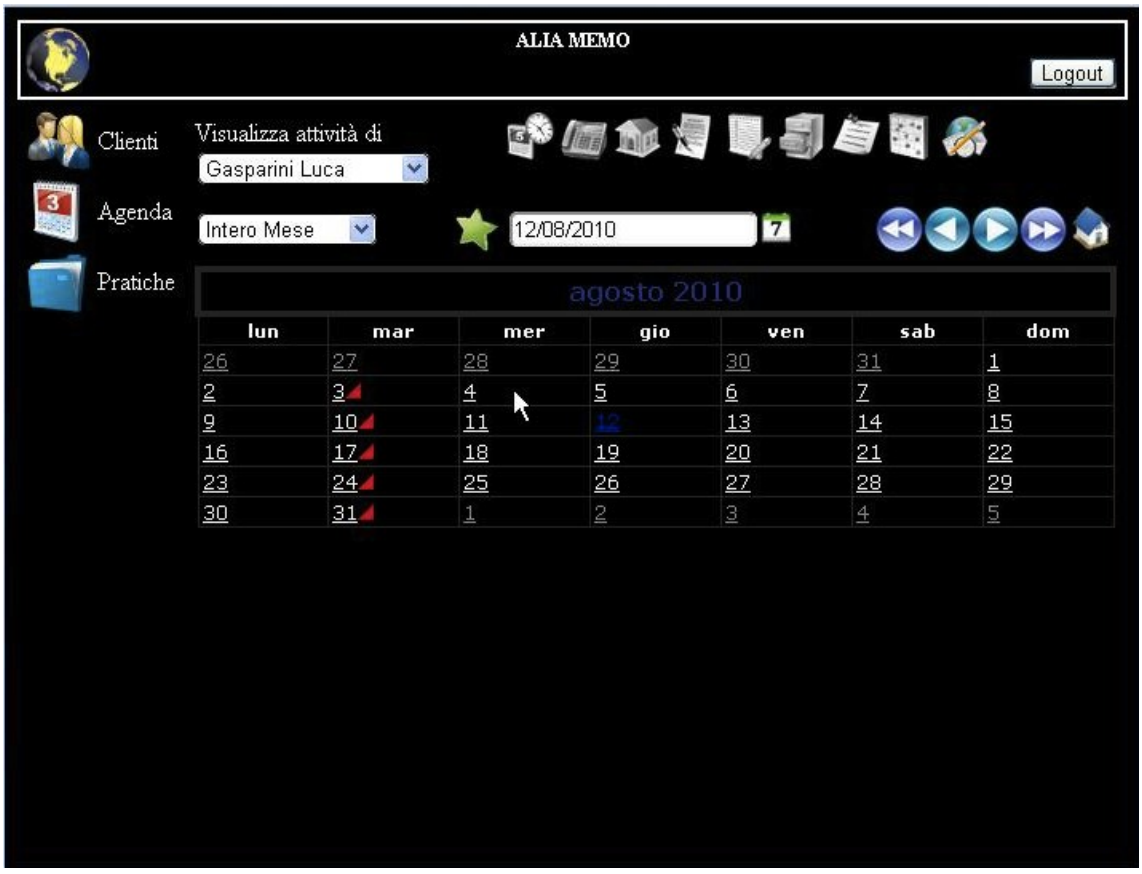
- visualizzazione “mese”
- visualizzazione “giorno”

Di default viene mostrata la visualizzazione “giorno” che mostra tutti gli eventi della giornata.



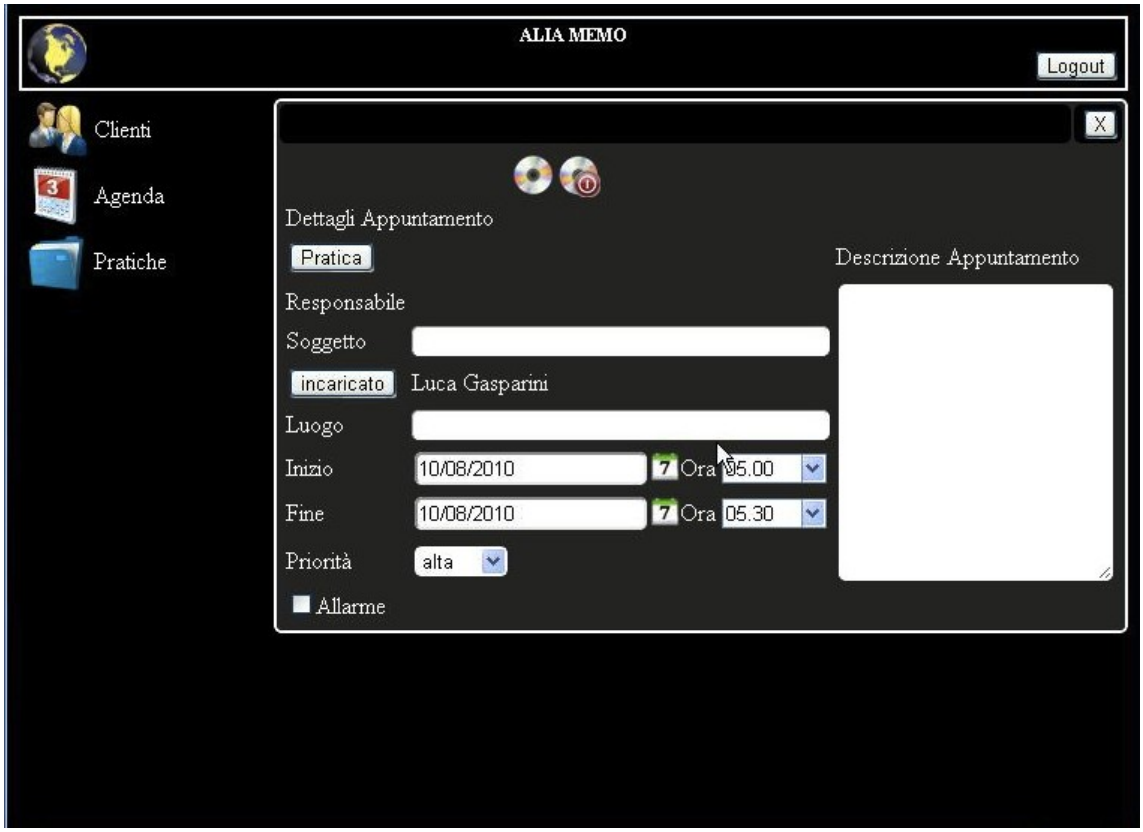
Ogni riquadro rosso rappresenta un evento ed è posizionato accanto all'altezza delle ore in cui si svolge l'evento. Cliccando sul riquadro è possibile accedere al dettaglio dell'evento selezionato. Tramite la drop “Visualizza Attività di” è possibile filtrare gli eventi in base alla persona incaricata di adempiere a tale impegno. Cliccando in uno dei pulsanti appena a destra è invece possibile filtrare gli eventi in base al tipo. Il calendario al di sotto dei pulsanti filtro permette di scegliere il giorno da visualizzare, che può essere cambiato anche utilizzando i tasti alla sua

destra. Cliccando su una delle caselle contenenti un orario, si passa alla creazione di un evento fissato per quel giorno e per l'ora selezionata. La drop “singolo giorno”/”intero mese” permette di scegliere la modalità di visualizzazione. Scegliendo “intero mese” si accederà alla pagina di figura in cui sono visualizzati tutti i giorni del mese. I giorni in cui è presente un evento sono contrassegnati con un simbolo rosso. Passando con il mouse sopra il simbolo rosso è possibile vedere un tooltip con il tipo di evento e il titolo. Cliccando su un giorno qualunque si passa alla visualizzazione del giorno selezionato.



In base ai permessi di cui dispone un particolare utente, sarà possibile vedere tutti gli eventi, o solo quelli di cui è responsabile o incaricato. Oltre ai permessi di visualizzazione ogni utente avrà anche dei particolari permessi di modifica, che permetteranno di modificare solo alcuni campi dell'evento o solo alcuni eventi in particolare.

Il tasto a forma di stella verde, in entrambe le visualizzazioni, permette di creare un nuovo evento. Se è selezionato uno dei tasti filtro verrà creato un evento del tipo corrispondente, altrimenti verrà chiesto che tipo di evento creare. Le pagine di creazione/dettaglio di un evento sono tutte molto simili tra loro, con solo qualche campo di differenza. Qui di seguito verranno illustrati solo alcuni tipi di pagina di creazione/dettaglio:



il campo incaricato viene caricato di default con il nome dell'utente che sta creando l'evento, ma può essere cambiato cliccando sull'apposito pulsante.

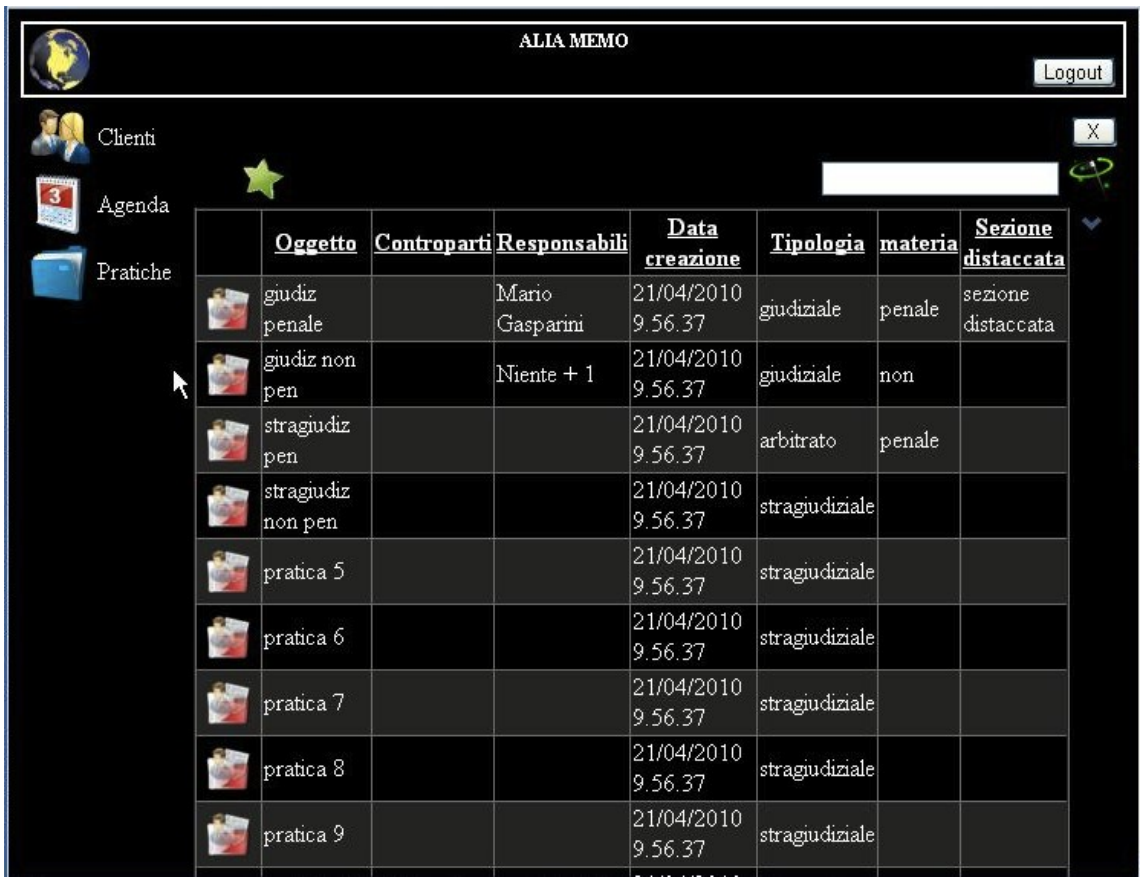
Il pulsante “pratica” permette di visualizzare lo storico delle pratiche e scegliere a quale di esse associare l'evento. Il responsabile dell'evento sarà il responsabile della pratica alla quale viene associato. È possibile impostare un allarme nel giorno in cui avviene l'evento, scegliendo l'ora in cui si vuole essere avvertiti. Per confermare e creare l'evento basta premere il tasto “salva”. Se si preme il tasto “salva ed esci”, dopo il salvataggio verrà mostrata l'agenda in modalità visualizzazione “giorno”.

Le scadenze ripetitive sono eventi che si ripetono con una certa cadenza.

Dalla pagina di inserimento è possibile scegliere con che cadenza l'evento si ripete. Se si sceglie “giornaliera” sarà possibile scegliere in quali giorni della settimana si ripeterà l'evento. Le altre possibilità sono “mensile”, “annuale” e “personalizzata”. Nel caso si selezioni la cadenza “personalizzata” è possibile scegliere ogni quanti giorni/ mesi/ anni far ripetere l'evento.

5.2.6 Pratiche

Cliccando sul tasto “pratiche” della barra laterale, si apre la pagina di visualizzazione dello storico delle pratiche. In questo caso, a differenza dello storico clienti, nella demo è presente solo la funzionalità di visualizzazione, mentre non è possibile creare, modificare o eliminare una pratica.



	Oggetto	Controparti	Responsabili	Data creazione	Tipologia	materia	Sezione distaccata
	giudiz penale		Mario Gasparini	21/04/2010 9.56.37	giudiziale	penale	sezione distaccata
	giudiz non pen		Niente + 1	21/04/2010 9.56.37	giudiziale	non	
	stragiudiz pen			21/04/2010 9.56.37	arbitrato	penale	
	stragiudiz non pen			21/04/2010 9.56.37	stragiudiziale		
	pratica 5			21/04/2010 9.56.37	stragiudiziale		
	pratica 6			21/04/2010 9.56.37	stragiudiziale		
	pratica 7			21/04/2010 9.56.37	stragiudiziale		
	pratica 8			21/04/2010 9.56.37	stragiudiziale		
	pratica 9			21/04/2010 9.56.37	stragiudiziale		

Come si vede dall'immagine infatti, sulla tabella non è presente il tasto elimina. Il resto dei comandi e delle funzionalità è equivalente a quelli visti per lo storico clienti.

6 CONCLUSIONI

L'obiettivo principale del lavoro di stage svolto presso ICHI S.R.L. era quello di studiare le diverse tecnologie attualmente disponibili per la realizzazione di un'interfaccia che permetta di accedere ad un gestionale per studi legali, precedentemente sviluppato dall'azienda, utilizzando un qualsiasi dispositivo mobile. Dopo aver analizzato diverse possibilità si è scelto di sviluppare una web application: questa soluzione ha il vantaggio che, l'applicazione risultante, sarà accessibile da un qualsiasi dispositivo in grado di accedere alla rete e dotato di un browser web. Tra le varie soluzioni disponibili per la realizzazione di un'applicazione web si è scelto di utilizzare quella proposta da Microsoft con il suo ASP.NET, questo perché l'azienda ha già una notevole esperienza nello sviluppo di applicazioni attraverso gli strumenti della piattaforma .NET. Nel primo periodo di stage si è preso confidenza con ASP.NET ed in particolare con il linguaggio di programmazione C#, dopodiché si è passati allo sviluppo vero e proprio dell'applicazione. Il risultato ottenuto al termine del periodo di stage è una demo che ha permesso di capire le potenzialità e i limiti della tecnologia utilizzata. In futuro sarà possibile completare l'applicazione integrando le funzionalità non ancora presenti e migliorando l'applicazione dal punto di vista dell'usabilità e dell'interfaccia grafica.

BIBLIOGRAFIA

1. http://quickstarts.asp.net/QuickStartv20/aspnet/Default.aspx
2. http://msdn.microsoft.com/
3. http://blog.taragana.com/index.php/archive/openssl-how-to-create-self-signed-certificate/it/
4. http://www.dia.unisa.it/~ads/corso-security/www/CORSO-9900/wap/index.htm
5. http://b62.tripod.com/wap/wapp.htm
6. http://www.openssl.org/
7. http://wikipedia.org/
8. http://www.kendar.org/iis-openssl.html
9. Murach's ASP.NET 3.5 Web Programming with C# 2008

RINGRAZIAMENTI

Desidero innanzitutto ringraziare l'Ing. Michele Moro per essersi dimostrato sempre disponibile a supportarmi e seguirmi durante la stesura di questa tesi. Un dovuto ringraziamento va all'Ing. Manuel Sgargetta per avermi dato la possibilità di svolgere questo tirocinio all'interno della sua azienda e per i suoi preziosi consigli. Grazie anche al dott. Luca Baseggio per l'aiuto che mi ha dato durante la realizzazione di questo lavoro. Ringrazio inoltre tutti i miei colleghi e compagni di studio (non faccio nomi per non dimenticare nessuno) per tutte le giornate passate a Treviso e per avermi aiutato a trovare la voglia di studiare nonostante tutte le distrazioni presenti in aula studio a giurisprudenza e in laboratorio di informatica. Un ringraziamento speciale va sicuramente ai miei amici Alessandro, Giacomo, Riccardo e Stefano; e qui non credo serva mettere una motivazione in particolare. Infine il ringraziamento più importante va alla mia famiglia: i miei genitori Mario e Nila e mio fratello Nicola, per avermi permesso, con il loro aiuto, non solo economico ma anche morale, di raggiungere questo obiettivo.

Grazie.