Ministero dell'Istruzione, dell'Università e della Ricerca Servizio Automazione Informatica e Innovazione Tecnologica

Modulo 16

Creazione e mantenimento di un sito Web

ForTIC

Piano Nazionale di Formazione degli Insegnanti sulle Tecnologie dell'Informazione e della Comunicazione

Percorso Formativo C

Materiali didattici a supporto delle attività formative

2002-2004

Promosso da:
- Ministero dell'Istruzione, dell'Università e della Ricerca, Servizio Automazione Informatica e Innovazione Tecnologica
- Ministero dell'Istruzione, dell'Università e della Ricerca, Ufficio Scolastico Regionale della Basilicata

Materiale a cura di:
- Università degli Studi di Bologna, Dipartimento di Scienze dell'Informazione
- Università degli Studi di Bologna, Dipartimento di Elettronica Informatica e Sistemistica

Editing:
- CRIAD - Centro di Ricerche e studi per l'Informatica Applicata alla Didattica

Progetto grafico:
- Campagna Pubblicitaria - Comunicazione creativa

In questa sezione verrà data una breve descrizione del modulo.

Gli scopi del modulo consistono nel mettere in grado di:

- Conoscere le procedure necessarie per ottenere un dominio *Internet*, la registrazione di un sito e la notificazione a motori di ricerca.
- Installare e configurare il *software* per la gestione di un sito *Web*, utilizzazione di tale strumento per la creazione e l'aggiornamento del sito stesso.
- Implementare appropriate misure di sicurezza.

Il modulo è strutturato nei seguenti argomenti:

- **Attivazione**
  - Descrivere il processo per l'ottenimento di un dominio *Internet*.
  - Registrare il sito *Internet*.
  - Notificare a motori di ricerca esterni il sito *Web*.
- **Strumenti di gestione**
  - Confrontare gli strumenti attualmente disponibili di gestione di un sito *Web*.
  - Installare e configurare *software* per la gestione di un sito *Web*.
  - Creare e aggiornare un sito *Web* usando strumenti di gestione.
- **Sicurezza**
  - Implementare appropriate misure di sicurezza in un sito *Web*.
  - Usare e valutare i risultati di uno strumento di memorizzazione delle visite al sito.

Introduzione

Attivazione

Cosimo Laneve

Il processo per l'ottenimento di un dominio Internet

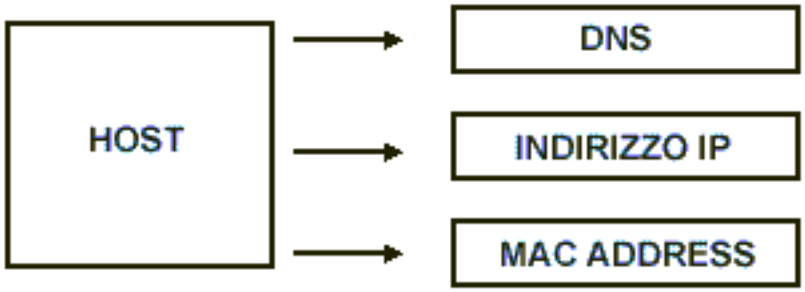
I nomi di dominio

Benché gli indirizzi IP forniscano una rappresentazione conveniente e compatta per specificare la provenienza e la destinazione nei pacchetti inviati attraverso *Internet*, gli utenti preferiscono assegnare alle macchine dei nomi facili da ricordare, che riflettano di solito il nome di una azienda o di un ente o di una persona fisica: i **nomi di dominio**. Questi nomi, a differenza dei numeri IP, detti a basso livello, sono nomi ad alto livello, e come i nomi IP sono unici: nessuno su *Internet* può avere lo stesso nome di dominio di un'altra persona. Su *Internet* ci sono opportune macchine (i DNS, **Domain Name Server**) che, quando interrogati, restituiscono l'indirizzo IP corrispondente a un nome di dominio, e viceversa il nome di dominio corrispondente all'indirizzo IP.

Indirizzi IP per accedere ai servizi

Il metodo di denominazione è interessante per due motivi: primo, esso è stato usato per assegnare nomi di macchine in tutta l'*Internet*; secondo, l'implementazione del meccanismo di conversione dei nomi fornisce un esempio su larga scala del paradigma *client-server* descritto nei capitoli precedenti, perché impiega un insieme di ***server*** distribuito geograficamente per porre in corrispondenza i nomi e gli indirizzi.

La forma dei nomi ad alto livello è importante perché determina il modo in cui i nomi vengono convertiti in nomi di livello inferiore o associati ad oggetti, nonché il modo in cui le assegnazioni dei nomi vengono autorizzate. Quando soltanto poche macchine sono interconnesse, la scelta dei nomi è semplice, e qualsiasi forma andrà bene. In *Internet* sono connesse oltre centomila macchine per cui la scelta di nomi simbolici diventa molto difficile.



nomi a basso ed alto livello

Gli organismi internazionali e nazionali per la gestione dei nomi di dominio

Nella scelta del nome di dominio, ad esempio XXX.YYY.ZZZ, occorre ricordare che essi sono composti da diverse parti. L'estensione ZZZ è la parte che identifica la nazione dove è mantenuto il domino oppure il tipo di contenuti del sito. Tipiche estensioni sono: .it, .com, .net, .org. Questi sono i nomi di primo livello, gestiti da un apposito organo internazionale, lo IANA (*Internet Assigned Number Authority*).

Quest'organo delega le autorità nazionali per la gestione e la regolamentazione degli indirizzi di secondo livello YYY. Gli organismi nazionali per esempio controllano che non ci siano omonimie nei nomi di secondo livello, gestiscono l'indirizzamento verso e dai nomi di secondo livello. Il processo appena descritto continua con i domini di terzo livello XXX, per i quali il riferimento diventa il possessore del dominio di secondo livello, e la sua gestione è demandata al *system manager* della rete locale.

In Italia, l'ente che gestisce il dominio it è il **GARR** (Gruppo di Armonizzazione delle Reti di Ricerca) che fa capo al CNR, e quindi al Governo. Negli Stati Uniti la gestione aveva un organo corrispondente al **GARR** fino a qualche tempo fa. Quando ci si è resi conto che *Internet* poteva diventare una forma di lucro, la gestione è stata privatizzata ed assegnata a società concessionarie che impongono tariffe annuali.

Indicazioni e moduli per la richiesta di nuovi domini

Tutte le indicazioni del processo per ottenere un nuovo dominio *internet* di secondo livello in Italia, compresi i moduli necessari alla richiesta di registrazione, sono disponibili alla URL http://www.nic.it/RA. Le riassumiamo brevemente di seguito.

Se si registra un dominio con estensione .it, si deve inviare (via fax allo 050/542420) alla *Registration Authority* Italiana la LAR (Lettera di Assunzione Responsabilità) che è disponibile *on-line*. La *Registration Authority* Italiana controlla che la lettera di assunzione di responsabilità sia stata correttamente compilata e firmata, notificando via *e-mail* al *provider/maintainer*, il cui **tag** è riportato nel corpo della lettera, il ricevimento della stessa. Inoltre, lo invita ad inviare il modulo elettronico necessario a completare la registrazione. Una volta superati con esito positivo tutti i controlli formali e tecnici, la *Registration Authority* provvede alla registrazione del nome a dominio e al suo caricamento nel Registro dei Nomi Assegnati.

Registrare il sito Internet

La registrazione di un nuovo dominio, solitamente, impiega 1-2 giorni, e la trasmissione della stessa tra i 2 e 3 giorni. La trasmissione è il processo con il quale il *provider* automaticamente aggiorna le proprie registrazioni (le tabelle DNS) per mostrare la nuova informazione.

La registrazione con l'estensione .it è limitata ad aziende e professionisti che possano dimostrare la loro attività tramite certificato di iscrizione ai registri IVA o certificato di iscrizione della Società alla Camera di Commercio, come richiesto dalla *Registration Authority* Italiana. Il servizio fornito dalla *Registration Authority* è il mantenimento del nome a dominio nel registro dei nomi assegnati, per il periodo di un anno. Dal 1 gennaio 2003 la tariffa unitaria addebitata al *provider/maintainer* è intorno ai 5 euro.

Per estensioni come .org, .net, .com, la registrazione è subordinata al pagamento di 35 dollari all'anno (i primi due anni vanno pagati in anticipo) che verranno addebitati direttamente da *InterNIC*. Una volta comunicata dalla *Registration Authority* competente l'avvenuta registrazione, il sito sarà raggiungibile con il proprio URL.

Notificare a motori di ricerca esterni il sito Web

La notifica delle pagine *Web* del proprio sito negli archivi dei motori di ricerca, può avvenire in due modi: sia attraverso la registrazione manuale, tipicamente da parte del responsabile del sito, che automaticamente mediante opportuni *software* che riescono a visitare milioni di siti *Web* al giorno, inserendo le nuove pagine nell'indice dei motori di ricerca ed aggiornando le informazioni su quelle già censite. Attraverso questo aggiornamento automatico, i motori di ricerca mantengono un archivio. Difficilmente sarà mai possibile l'intero *Web*. Basti pensare che il numero totale di pagine censite nel 2001 da **AltaVista** è di 550 milioni, quello di ***Google*** (il motore con l'archivio più completo) è di 1.3 miliardi, contro una stima di 5 miliardi di pagine *Web* e 550 miliardi di documenti che sembra componevano l'intera Rete a Marzo 2001.

Ad esempio, in *Google* è possibile inviare direttamente l'URL attraverso la pagina (**http://www.google.it/intl/it/add_url.html**). Per segnalare un sito, è necessario inserire l'URL completo, compreso il prefisso http://; ad esempio: http://www.google.com/. È possibile anche aggiungere commenti o parole chiave che descrivano il contenuto della pagina.

Per ogni pagina *Web* referenziata dai motori di ricerca, viene memorizzato parte del testo in essa contenuto, in modo tale che, ad ogni ricerca dell'utente, viene presentata una lista delle pagine *Web* dove figurano le parole che interessano.

Strumenti di gestione

Cosimo Laneve

Il Web Management

Internet esisteva già da oltre vent'anni, quando, poco dopo la sua nascita, il *Web* cominciò ad aprirsi alle attività commerciali. Allora, la presenza sulla rete era più o meno equivalente a un manifesto informativo sull'azienda. La gestione di un sito era del tutto simile a quella di un cartellone pubblicitario. Quest'ultimo va soprattutto disegnato con cura, ma una volta stabilitone il *layout*, i contenuti e dove piazzarlo, il gioco è fatto. *Web Management*, allora, significava soprattutto ideare un *look* accattivante, preoccuparsi che le informazioni fossero chiare e aggiornate. Ma l'aggiornamento poteva essere effettuato anche con una periodicità costante e lasca, poiché, in molti casi, le informazioni, scarse e istituzionali, variavano poco.

L'evoluzione del Web management

Ben presto, però, ci si è resi conto che il *Web* offriva maggiori potenzialità di un semplice cartellone posto sul bordo della carreggiata delle autostrade informatiche. Soprattutto perché queste ultime si andavano trasformando nel villaggio globale. Da cartellone a vetrina, da vetrina a negozio, il sito commerciale è evoluto rapidissimamente. Altrettanto velocemente sono evolute le problematiche che bisogna gestire:

- aggiornamento, accessibilità e disponibilità delle informazioni;
- visibilità e diffusione del sito;
- affidabilità, prestazioni e sicurezza del sistema.

Naturalmente, il sito-negozio è solo un esempio, ma per *e-commerce* si può intendere qualcosa di più complesso. Se poi consideriamo un uso interno della *Web technology*, ci rendiamo conto che le problematiche di gestione diventano molto più sofisticate. Prima si trattava di creare una *intranet* e dopo un'*extranet* collegandosi all'*intranet* dei *partner* e, di disegnare una strategia di *e-business*: il vecchio CED, deve rispondere alle esigenti richieste con un *budget* limitato, risorse umane impreparate e insufficienti e, spesso, senza avere la conoscenza necessaria per avviare lo sviluppo dei progetti. Si è compresa l'importanza dell'approccio *Internet* dell'azienda, ma troppo frequentemente vengono sottovalutate le difficoltà gestionali e la complessità

dell'infrastruttura che questo comporta, soprattutto in termini di impatto sull'intero sistema informativo, da un lato, e sui processi di *business*, dall'altro.

Il sistema informativo diventa infrastruttura

Il sistema informativo è da tempo, ormai, un servizio indispensabile per l'attività di *business*. Nella nuova era, è il *Web* ad assumere questo ruolo centrale. Per cui la gestione di quella che è diventata, a tutti gli effetti, un'infrastruttura unica, deve cambiare da gestione dei vari componenti a servizi di gestione. Un'infrastruttura che diventa così importante ed estesa trasversalmente in azienda, rende impossibile, confinare i vari interventi gestionali all'interno di una singola area. Ma anche le suddivisioni di livello dipartimentale o funzionale tendono a saltare, con la creazione di gruppi di lavoro virtuali che interessano più reparti. In una situazione del genere, è difficile garantire i livelli di servizio.

La struttura del sistema informativo

Gartner Group prevede che il 50% delle aziende con siti *Web* votati all'*e-business* non riusciranno a raggiungere le capacità gestionali necessarie entro il 2004, a meno che inizino a pensare in termini di servizi di gestione, definendo ruoli, responsabilità e collegamenti tra i domini di gestione. Vediamo rapidamente quali domini occorre considerare per il *Web management*:

- **Controllo dei contenuti:** ottimizzare l'uso delle risorse e aumentare la produttività sono due delle ragioni che spingono al controllo dei contenuti. Gli strumenti per tali controlli sono *proxy*, ***firewall*** e sonde. È importante che i limiti siano noti in azienda, per rispettare la *privacy*. Senza esagerare, perché ne va della soddisfazione del personale. Inoltre, non bisogna confondere il controllo dei contenuti con una procedura di sicurezza.
- **Misura del tempo di risposta:** quando si attraversa *Internet* i tempi di risposta diventano praticamente casuali. Soprattutto dipendono da troppi fattori: capacità di banda, traffico sulla linea, stato di questa stessa, numero di accessi al *Web* ***server***, potenza di quest'ultimo. Tra i vari approcci: misura presso il *client*, *client simulation*, *application analyzer*.
- ***Web server runtime:*** l'uso del *Web server* va ottimizzato con strumenti

aggiuntivi. I *load balancer*, che permettono di distribuire le richieste dirette a un servizio *Web* su più ***server*** multipli. Gli strumenti di monitoraggio permettono di verificare la percezione del servizio da parte dell'utente. Ci sono i *tool* di analisi dei guasti e i *Web log analyzer*.

Concludiamo osservando che è necessario istituire in azienda uno *staff* per lo sviluppo del *Web*, che includa i tecnici specializzati, ma che sia verticale in tutta l'azienda.

Confrontare gli strumenti attualmente disponibili di gestione di un sito Web

Quando si lavora con un sito *Web*, cioè si modificano il progetto, la relazione tra le pagine, i contenuti delle pagine, è tipico che manchino alcuni *hyperlink* tra pagina e pagina, manchino titoli, che il caricamento delle pagine sia lento. Quando ciò accade e quando il numero di pagine che il sito gestisce inizia a diventare rilevante, allora diventa veramente difficile trovare difetti manualmente e risolverli. Da qui la necessità di avere strumenti di gestione di siti *Web* che consentano di ridurre i costi e gli inconvenienti della gestione manuale, e di realizzare ottimizzazioni.

Se si perseguono risultati editoriali professionali, dunque complessi e con effetti grafici avanzati, un buon strumento di gestione di siti *Web* dovrebbe:

- consentire di creare e gestire siti *Web* di alta qualità professionale, possibilmente con meccanismi WYSIWYG;
- **identificare** *hyperlink* mancanti, **o** datati, ***and*** altri problemi di contenuto **delle pagine**;
- rilevare le **pagine che non hanno elementi cruciali**, come i ***tag*** *Title*, *Metadata*, e gli attributi *image*;
- riportare, pagina per pagina, i problemi che sono stati rilevati.

I tre strumenti di gestione di siti più diffusi sono ***Adobe GoLive***, ***Microsoft FrontPage***, e ***Macromedia Dreamweaver***. Discuteremo in dettaglio il primo, che è quello più ricco di funzionalità; il secondo è stato discusso lungamente nel **modulo 13**, ed in questo modulo ci occuperemo di aspetti avanzati che riguardano la gestione collaborativa del sito; accenneremo soltanto a *Macromedia* ***Dreamweaver***.

Adobe GoLive

(http://www.adobe.com)

GoLive è un programma per la creazione di siti mediante un *editor*, funzioni di gestione che consentono di sviluppare siti *Web* a livello professionale con immagini, suoni e animazioni. Esso include *Adobe Web Workgroup Server*, che, come vedremo semplifica la collaborazione tra più *Web designer*.

Le principali caratteristiche di *Adobe* ***GoLive*** sono:

- **progettazione di diagrammi del sito:** è possibile creare un sito e sviluppare un diagramma di progettazione da sottoporre per eventuali commenti e approvazioni. I diagrammi consentono, in particolare, di visualizzare le relazioni e i collegamenti esistenti tra le pagine. Dopo aver

eseguito il diagramma del sito, è possibile generare velocemente una mappa del sito (mediante la funzione del sommario) e trasformare il diagramma in pagine interattive per il sito.

- **Creazione e progettazione visiva delle pagine:** la produzione di pagine è facilitata, senza che sia necessario elaborare il codice. È possibile salvare, gestire e applicare stili di testo con una opportuna funzione Tavolozza Stili HTML. Con questa funzione, è possibile creare nuovi stili basati sul testo formattato di una pagina e applicarli al testo all'interno di tutto il sito; è anche possibile definire, applicare e visualizzare gli *Style Sheet* a cascata. Inoltre è possibile visualizzare a schermo diviso sia il *layout* della pagina che del codice sorgente. Infine è definita una procedura guidata per il sito per creare e importare altri siti.
- **Integrazione degli strumenti *standard* dell'industria nel flusso di lavoro:** è possibile elaborare immagini, elementi grafici e applicazioni di animazione già esistenti e integrarli all'interno di ***GoLive***. I *file* originali, non ottimizzati, si possono posizionare nelle pagine e convertirli in seguito in immagini grafiche ottimizzate e formattate per il *Web*. È anche possibile aggiungere animazioni *Smart* SWF alle pagine *Web*, e sviluppare contenuto multimediale elaborato per il *Web*.
- **Pacchetti con azioni *JavaScript* predefinite:** *Adobe* ***GoLive*** contiene un pacchetto di 14 azioni *JavaScript* con cui è possibile aggiungere funzioni di interattività alle pagine *Web* che si stanno creando senza dover occuparsi del tipo di programmazione necessario.

Microsoft FrontPage

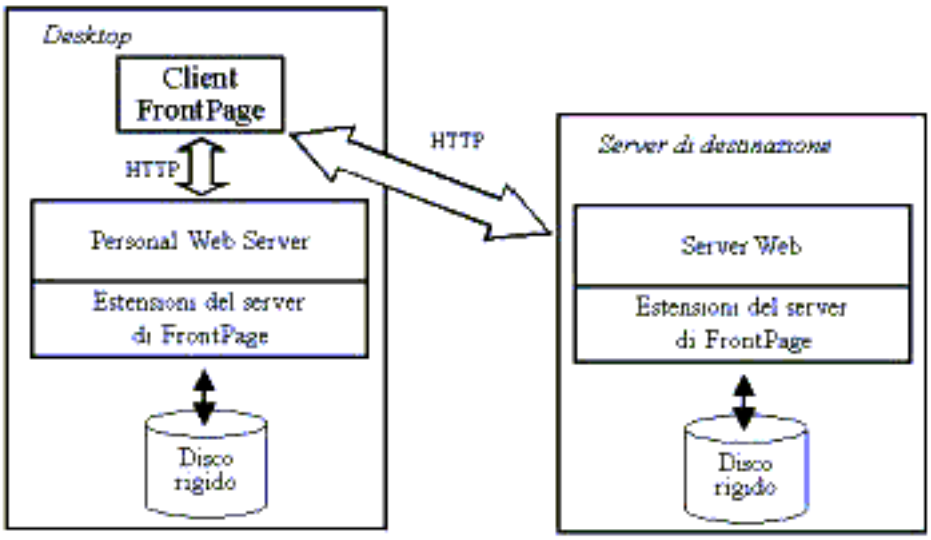
(http://www.microsoft.com/frontpage/)

Di *Microsoft* ***FrontPage*** abbiamo discusso nel modulo 13. Qui ci concentriamo su aspetti avanzati per la gestione di siti *Web* complessi. Questo servizio è fornito da estensioni di ***FrontPage*** che sono eseguiti sul ***server*** *Web* (come lo ***SharePoint Team*** che analizzeremo in seguito). Queste estensioni supportano le seguenti funzioni:

- Modifica dei *Web* di ***FrontPage***. Se ad esempio si sposta una pagina da una cartella a un'altra dello stesso *Web* di ***FrontPage***, le estensioni del ***server*** aggiorneranno automaticamente tutti i collegamenti ipertestuali alla pagina spostata eventualmente contenuti in altre pagine o documenti di *Microsoft Office* presenti nel *Web* di ***FrontPage***. L'aggiornamento viene effettuato direttamente sul *server Web*.
- Amministrazione dei *Web* di ***FrontPage***. L'amministratore di un *Web* di ***FrontPage*** ha ad esempio la possibilità di specificare i nomi degli utenti autorizzati ad amministrare, modificare o esplorare il *Web*.
- Esplorazione dei *Web* di ***FrontPage***. Se ad esempio gli utenti di un *Web* di ***FrontPage*** partecipano a un gruppo di discussione, le estensioni del ***server*** aggiorneranno l'indice dei collegamenti ipertestuali agli articoli relativi alla discussione, ad altri temi, a sommari e a moduli di ricerca per consentire di individuare rapidamente le pagine di maggiore interesse.

Quando si utilizza ***FrontPage*** *Explorer* per aprire un *Web* situato su un ***server*** che dispone delle estensioni di ***FrontPage***, tutte le informazioni relative al *Web* di

FrontPage, ad esempio la mappa dei collegamenti, vengono copiate nel *computer client* in modo da poter essere visualizzate, mentre l'insieme delle pagine e dei *file* che costituiscono il vero e proprio *Web* di ***FrontPage*** rimangono sul *server Web*. Le pagine vengono scaricate da *Internet* solo quando vengono aperte in ***FrontPage*** *Editor* per essere modificate. Questo sistema è molto efficiente poiché consente la modifica di siti *Web* direttamente sul ***server*** scaricando e modificando un unico *file*.



Interazione tra cliente FrontPage e Server con estensioni FrontPage

Quando un *server Web* dispone delle estensioni di ***FrontPage***, le operazioni di modifica e di amministrazione dei *Web* possono essere effettuate utilizzando qualsiasi PC o *computer Macintosh* che esegua il *software client* di ***FrontPage*** e sia connesso a *Internet* o a una rete *Intranet* locale. Invece le funzioni di esplorazione sono accessibili a qualsiasi *browser Web* connesso a *Internet* o a una rete *Intranet*, visto che le comunicazioni fra il *computer client* e un *server Web* con le estensioni di ***FrontPage*** avvengono tramite il protocollo HTTP.

Alcune delle funzioni supportate dalle estensioni del ***server*** di ***FrontPage*** sono:

- **Creazione e gestione di mappe complete dei collegamenti ipertestuali contenuti nei *file* di un *Web* di *FrontPage***. Tali mappe vengono utilizzate per la visualizzazione dei collegamenti ipertestuali in ***FrontPage***. Se un *Web* viene copiato da un ***server*** a un altro, la corrispondente mappa dei collegamenti ipertestuali verrà rielaborata.
- **Indicizzazione completa del testo di tutte le pagine di un *Web***. Questa funzione consente agli utenti finali di ricercare in un *Web* pagine contenenti termini o frasi specifiche.
- **Creazione e gestione di una struttura per la creazione e la riorganizzazione di un *Web***. Nella struttura, che può essere visualizzata, creata o modificata in ***FrontPage***, vengono definite le pagine principali di un *Web* e le relazioni esistenti tra esse. Se si modifica la struttura di un *Web*, le pagine interessate verranno aggiornate di conseguenza.
- **Applicazione di temi ai *Web***. Un tema è un insieme di oggetti grafici e stili

utilizzati per la visualizzazione degli elementi delle pagine, ad esempio i colori dello sfondo e del testo, i punti elenco, i bordi e le linee orizzontali, con colori e stili coordinati per consentire la creazione di *Web* dall'aspetto uniforme e accattivante. Se si applica un tema a un *Web*, tutte le pagine verranno aggiornate automaticamente.

- **Creazione e gestione di Elenchi attività in cui vengono specificate le operazioni da svolgere per completare un *Web***. Le attività sono collegate alle pagine a cui si riferiscono.
- **Gestione di autorizzazioni separate per ciascun *Web***. È possibile specificare gruppi di amministratori, autori e utenti finali distinti per ciascun *Web* di ***FrontPage***.

Macromedia Dreamwear

(http://www.macromedia.com)

Come gli altri anche *Macromedia* ***Dreamweaver*** consente sia di visualizzare il *layout* delle pagine HTML che direttamente il codice, dando la possibilità di editare i due *layout* con un editore WYSIWYG. ***Dreamweaver*** consente la creazione di tabelle e *frame* mediante meccanismi di tipo *drag-and-drop*, consente l'utilizzo di *Cascading Style Sheet*, *JavaScript* e *Dynamic*. Alcuni meccanismi di gestione dei siti sono anche compresi, sebbene non troppo avanzati come gli altri due. Ad esempio la verifica e modifica dei collegamenti ipertestuali e una libreria di contenuti che sono solitamente utilizzati. ***Dreamweaver*** è disponibile sia su piattaforma *Macintosh* che *Windows*.

Installare e configurare software per la gestione di un sito Web

L'istallazione e configurazione dei *software* discussi in precedenza (*Adobe* ***GoLive***, *Microsoft* ***FrontPage*** e *Macromedia* ***Dreamweaver***) è piuttosto semplice ed è guidata. È sufficiente infatti inserire il disco di installazione per la propria piattaforma e seguire i passi specificati (tipicamente occorre rispondere sempre Si oppure Ok quando richiesto. Perciò non ci soffermiamo su questo aspetto più di tanto.

Creare e aggiornare un sito Web usando strumenti di gestione

In questa sezione analizzeremo gli strumenti messi a disposizione da *Adobe* ***GoLive*** e *Microsoft* ***FrontPage*** per la creazione e gestione collaborativa dei siti *Web* da parte di un gruppo di persone (*workgroup management*) piuttosto che da un singolo. Naturalmente questi aspetti, trascurati nei moduli precedenti, sono essenziali quando si deve sviluppare un sito complesso.

Adobe ***GoLive***

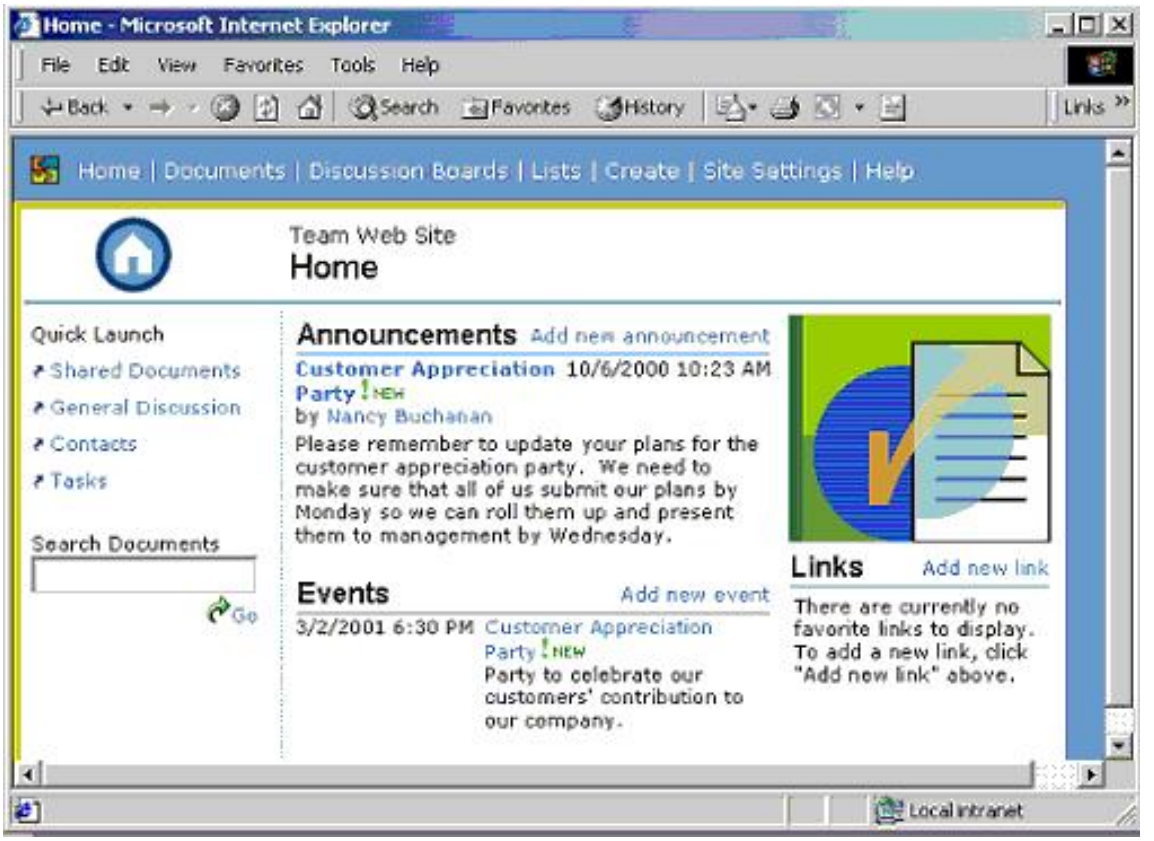
Adobe ***GoLive***, utilizzato con il *Web Workgroup Server*, fornisce una serie di strumenti per la progettazione, elaborazione, archiviazione, e gestione collaborativa di un sito. A tal fine, di seguito sono illustrati i passi da seguire per attivare questo meccanismo, e per attivare FTP *publishing server* che è alla base di esso.

- **Conversione del sito in un sito *workgroup***. Aprire il *file* del progetto del sito per visualizzare la finestra del sito, e scegliere ***Site > Workgroup > Convert to Workgroup Site***. Selezionare **OK** nella finestra di dialogo di conferma. Il processo sovrascrive i *file* del progetto con i nuovi *file* del sito *workgroup*. Alternativamente è possibile utilizzare il ***GoLive*** *Site Wizard*, per creare un nuovo sito *workgroup* a partire da quello esistente. In tal caso usare l'opzione ***GoLive*** *Site* e selezionare *Import*.
- ***Login* al *Web Workgroup Server***. Nella finestra di dialogo *Convert Actual Site To Workgroup Site*, introdurre il nome del *workgroup*, la propria *login* e *password*, il numero di porta, e quindi selezionare *Convert*. Chiedere all'amministratore del sito le informazioni relative alla propria *login*. Se si sta eseguendo il *Web Workgroup Server* sulla propria macchina locale, allora bisogna inserire *localhost* come *Server*, la propria *login* e *password*, ed inserire 1102 come numero della porta (o quello che è stato digitato durante l'installazione del *Web Workgroup Server*). In alcuni casi, può essere necessario inserire il proprio numero IP invece che *localhost*.
- **Isolare ed editare un *file***. A questo punto si può iniziare a lavorare sui singoli *file*. Per evitare che più persone modifichino al contempo lo stesso *file*, rendendolo inconsistente, occorre isolarlo quando qualcuno lo utilizza. A tal fine è sufficiente selezionare il *file* nella finestra dei *files*, e selezionare il bottone *Check Out* nella barra di controllo del *Workgroup*. A questo punto si può editare tranquillamente il *file*, ed una volta terminato, selezionare ***Edit***, quindi scegliere ***File > Save*** per salvare le modifiche.
- **Inserimento del *file***. Una volta terminata la fase editoriale, il *file* può essere inserito nel *Web Workgroup Server*, che automaticamente aggiunge una nuova versione ai *file* della lista di revisione. A tal fine occorre selezionare il bottone *Check In* nella barra degli strumenti, come mostrato nella figura sottostante, e ancora *Check In* nella finestra di dialogo.
- **Fasi di inizializzazione**. Quando il sito è pronto per essere accessibile, occorre inizializzare il ***server*** FTP che ospita il sito. A tal fine, selezionare *Site > Workgroup > Open Workgroup Administration* e quindi entrare in *Web Workgroup Server Administration*. Per far ciò bisogna possedere diritti di amministratore. Nella finestra *Web Workgroup Server Administration* scegliere *Sites* nel pannello di sinistra, selezionare il nome del proprio sito dalla lista del pannello di destra e quindi selezionare *New Publish Server* nel pannello di sinistra. A questo punto ci viene richiesto di inserire il nome del *Server* e del nodo FTP, e su richiesta, la cartella dei *file*. Quindi si clicca su *Save* e poi su *Log off* nell'angolo in alto a destra della finestra di *Web Workgroup Administration*.
- **Pubblicazione del sito**. Una volta pronti a rendere disponibile il sito sulla rete, cliccare il bottone *Publish Server Connect/Disconnect* nella barra degli strumenti come mostrato in Figura per connettersi al ***server***. Una volta connessi, il sito può essere pubblicato con gli strumenti di ***GoLive***.

Microsoft Front Page (il servizio SharePoint Team)

Discutiamo ora di alcuni aspetti di gestione di gruppo, che viene fornita dal servizio *SharePoint Team*, una delle estensioni di ***FrontPage*** che è necessario sia installata sul

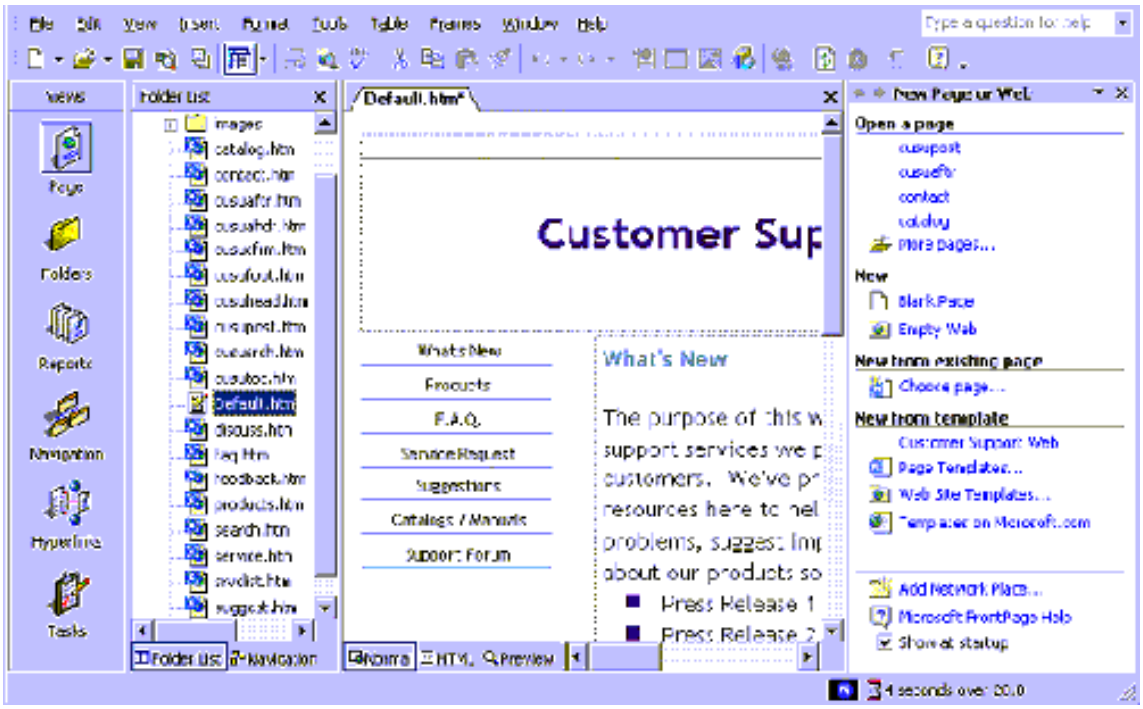
Web server. Questo applicativo consente ai partecipanti allo sviluppo del sito di trovare e condividere documenti, di discutere, di annunciare i prototipi, di analizzare le versioni intermedie. Inoltre, consente di sviluppare i siti in tutti i linguaggi supportati da ***FrontPage*** (circa una trentina al momento).



Una pagina iniziale di Microsoft Explorer

SharePoint si trova in una *directory* separata del CD di installazione di ***FrontPage*** o di quello di *Office* con ***FrontPage***. Una volta installato su un ***server*** che contiene *Microsoft* IIS (vedi moduli 13 e 17), è sufficiente accedere allo spazio del gruppo mediante il *browser Explorer*. Questi sono i passi necessari per creare lo spazio del gruppo:

- **Determinare dove si troverà il sito *Web*.** Ci sono due casi, a seconda che il sito si troverà su *Internet* o su una *Intranet* di una organizzazione. Se il sito si troverà su *Internet* allora è necessaria l'URL che potrà essere richiesta all'*Internet Provider* (è necessario che questi utilizzi tecnologia *Microsoft*). Nel secondo caso bisogna rivolgersi all'amministratore del sistema.
- **Creazione del sito.** In ***FrontPage***, accedere a *File*, puntare a *New* e quindi selezionare *Page or Web*. Nella colonna di destra che appare, sotto *New from template*, selezionare *Web Site Templates*. Quindi selezionare *SharedPoint-based Team Web Site*, e quindi in *Specify the location of the new Web* digitare l'URL del proprio sito *Web*. Nella lista dei *folder*, scegliere *Default.htm* che è l'*home page* del nuovo sito *Web*.



FrontPage con la colonna per i nuovi siti Web

Qui di seguito illustriamo alcune caratteristiche di *SharePoint*:

- **Editare il sito *Web* mediante *browser*.** *SharePoint* consente di creare i documenti, inserirli nel sito, partecipare alle discussioni, ricevere le notifiche di nuove pagine senza mai lasciare l'ambiente del *browser*. A tal fine è sufficiente selezionare il sito mediante il *browser* ed aggiungere l'annuncio selezionando *Add new announcement* e riempiendo la *form* relativa. Dopo aver cliccato su *Save and close*, è possibile visualizzare il proprio annuncio sulla *home page* del sito.
- **Discussioni e sottoscrizioni.** *SharePoint* consente di postare e replicare a commenti sulle pagine prodotte del sito, o in generale di *Internet*. Consente anche di abbonarsi per ricevere notifiche via *e-mail* quando ci sono nuove discussioni o nuove modifiche. A tal fine è sufficiente selezionare una qualunque pagina *Web*, quindi selezionare *Discuss* sui menù di *Explorer* (dalla versione 4 in poi), selezionare il bottone delle *Discussions* in basso a destra della finestra, e infine selezionare *Discussion Options*. Per aggiungere una nuova lista di discussione su *SharePoint*, cliccare su *Add* e poi su *Choose a discussion server* ed infine digitare l'url in *URL for your SharePoint team Web site*. A questo punto è possibile iniziare la discussione. A tal fine, bisogna cliccare su *Insert Discussion in the Document*. Nel caso la discussione non sia nuova, apparirà una pagina con diverse note. Da qui se ne può selezionare una e rispondere.
- **Libreria dei documenti.** *SharePoint* consente di creare e gestire una libreria di documenti che ne permette l'archiviazione. Gli utenti possono creare un nuovo documento per la libreria a partire da un modulo specifico, o possono modificare documenti pre-esistenti mediante il proprio *browser*.

È possibile anche ordinare con o senza opportuni filtri la libreria. Ecco alcune operazioni possibili:

- Dalla *home page* di *SharePoint* (nel *browser*), selezionare il *link Documents*. A questo punto è possibile sia creare una nuova libreria (selezionare *New Document Library*) o accedere ad una già esistente.
- Aprire un documento, quindi selezionare *Save as Web Page* dal menù *File*. Quindi bisogna digitare l'*URL for your SharePoint team Web site* e selezionare *Open*. A questo punto dovrebbe essere possibile vedere le librerie di documenti. Per salvare il documento in una libreria è sufficiente cliccare su *Save*. Per vedere il risultato di quanto fatto, andare nella *home page* di *SharePoint* e cliccare su *Documents*. Quindi se si seleziona la libreria appena salvata, sarà possibile vedere il documento nel *browser*.

Sicurezza

Cosimo Laneve

Implementare appropriate misure di sicurezza in un sito Web

Data la diffusione dei dispositivi che ospitano i siti *Internet*, è naturale attendersi che maggiore è la diffusione, e tanto più si corre il rischio che malintenzionati, conoscendo i limiti ed i difetti della tecnologia impiegata, tentino di forzarne la sicurezza. È di fondamentale importanza per un'azienda definire criteri di protezione specifici. Ad esempio bisogna definire come intervenire in caso di furto, se esiste una strategia di *backup*, e quali utenti hanno accesso a quali risorse facendo distinzione tra porzione pubblica e porzione privata del sistema. Bisogna comunque precisare che è importante valutare bene le impostazioni di sicurezza, in modo da identificare le specifiche più adeguate alla propria situazione aziendale, e non penalizzare eccessivamente le prestazioni. È chiaro che se si eccede nelle politiche di sicurezza si rischia di isolare il proprio sistema dal mondo esterno, rendendolo irraggiungibile ai più.

Poiché il sito *Web* poggia sul *Web server*, è chiaro che le problematiche di sicurezza riguardano principalmente il *Web server*. In questa parte del modulo analizziamo *Microsoft Internet Information Services* (IIS) che è stato introdotto nel **modulo 13**. Di problematiche di sicurezza (di altri sistemi) ci occuperemo anche nel **modulo 17**, quando discuteremo dei *server Web* in generale.

La sicurezza di Microsoft Internet Information Services

Qui descriviamo le procedure consigliate per configurare la sicurezza di un *server Web* che esegue *Microsoft Windows* e *Internet Information Services* (IIS). Essendo il *Web server* un processo, il quale viene eseguito all'interno del sistema operativo, è tacito che una buona sicurezza del sistema inizi proprio nel considerare l'interazione di IIS col sistema.

IPSec: è consigliabile impostare i criteri per il filtraggio di pacchetti IPSec (*Internet Protocol Security*) in ogni *server Web*. Questi criteri implementano un ulteriore livello di sicurezza in caso di violazione dei ***firewall***. È buona norma impostare più livelli di sicurezza: se ne fallisce uno, c'è sempre una seconda porta da forzare. In generale è necessario bloccare tutti i protocolli TCP/IP diversi dai protocolli che si desidera supportare e le porte da utilizzare. Per l'implementazione dei criteri IPSec, è possibile utilizzare lo strumento di amministrazione IPSec o lo strumento della riga di comando IPSecPol.

***Telnet*:** se si prevede di utilizzare il *server Telnet* incluso in *Windows*, è consigliabile specificare gli utenti che sono autorizzati ad accedere al servizio. A tale scopo, eseguire la procedura seguente:

- Avviare lo strumento Utenti e gruppi locali.
- Fare clic con il pulsante destro del *mouse* sul nodo Gruppo e scegliere Nuovo gruppo dal *menu* di scelta rapida.

- Nella casella Nome gruppo digitare >*TelnetClients*.
- Fare clic su Aggiungi e aggiungere gli utenti a cui si desidera consentire l'accesso ***telnet*** al *computer*.
- Fare clic su Crea e quindi su Chiudi.

Dopo l'aggiunta del gruppo *TelnetClients*, il servizio *Telnet* consente l'accesso al ***server*** **solo** agli utenti appartenenti al gruppo creato in precedenza.

Impostazione degli elenchi ACL appropriati nelle *directory virtuali*: Sebbene la procedura descritta di seguito vari a seconda dell'**applicazione**, è possibile definire alcune regole che permettono di evitare intrusioni comuni nel sistema. Ecco una serie di operazioni utili:

- **Elenchi ACL predefiniti consigliati in base al tipo di *file*:** Anziché impostare gli elenchi **ACL** per ogni *file*, è consigliabile creare nuove *directory* per ogni tipo di *file*, impostare gli **ACL** per le *directory* e consentire l'ereditarietà degli elenchi per i *file*. Una struttura di *directory* potrebbe essere simile alla seguente:
  - c:\inetpub\wwwroot\mioserver\static(.html).
  - c:\inetpub\wwwroot\mioserver\inclusione (.inc).
  - c:\inetpub\wwwroot\mioserver\script (.asp).
  - c:\inetpub\wwwroot\mioserver\eseguibili (.dll).
  - c:\inetpub\wwwroot\mioserver\immagini (.gif, .jpeg).

  È inoltre importante prestare particolare attenzione alle seguenti *directory*:
  - c:\inetpub\ftproot (***server*** FTP).
  - c:\inetpub\mailroot (***server*** SMTP).

  In entrambe queste *directory* l'elenco **ACL** è *Everyone* (Controllo completo) e deve essere sovrascritto con autorizzazioni più restrittive a seconda del livello di funzionalità del sistema in uso. Se si desidera supportare *Everyone* (Scrittura), spostare la cartella in un volume diverso da quello del ***server*** IIS oppure, tramite le quote disco di *Windows* 2000, limitare la quantità di dati che è possibile scrivere in queste *directory*.

Tipo di *file*	Elenchi di controllo di accesso (ACL>, *Access Control List*)
Processi *CGI* (.exe, .dll, .cmd, .pl)	*Everyone* (X) *Administrators* (Controllo completo) *System* (Controllo completo)
Inclusioni lato ***server*** (.asp)	*Everyone* (X) *Administrators* (Controllo completo) *System* (Controllo completo)
File di inclusione (.inc, .shtm, .shtml)	*Everyone* (X) *Administrators* (Controllo completo) *System* (Controllo completo)
Oggetti statici (.txt, .gif, . jpg, .html)	*Everyone* (R) *Administrators* (Controllo completo) *System* (Controllo completo)

- **Rimuovere ciò che non serve:** L'installazione delle applicazioni di esempio, che non devono essere mai installate in un ***server*** di produzione, non viene eseguita per impostazione predefinita. Alcune applicazioni

vengono installate in modo che siano accessibili solo da http://localhost, o 127.0.0.1. È comunque necessario rimuoverle.

Applicazione di esempio	*Directory* virtuale	Posizione
Applicazioni di esempi IIS	*IISSamples*	c:\inetpub\iissamples
Documentazione IIS	*IISHelp*	c:\winnt\help\iishelp
Accesso ai dati	\MSADC	c:\Programmi\Filecomuni\System\Msadc

Disabilitazione o rimozione di componenti COM non necessari: per la maggior parte delle applicazioni alcuni componenti COM non sono necessari e devono essere rimossi. In particolare, è consigliabile disabilitare il componente *File System Object*. Con questa operazione, tuttavia, viene rimosso anche l'oggetto *Dictionary*. È importante tenere presente che i componenti disabilitati potrebbero essere necessari per alcuni programmi. Ad esempio, *Site Server* 3.0 utilizza il componente *File System Object*, che è possibile disabilitare tramite il comando seguente:

regsvr32 scrrun.dll /u

Rimozione della *directory* virtuale IISADMPWD: la *directory* virtuale IISADMPWD consente di ripristinare la *password* di *Windows* NT e di *Windows* 2000. La *directory* è stata progettata principalmente per reti *Intranet*. Non viene installata automaticamente insieme a IIS 5 e non viene rimossa quando si aggiorna un ***server*** IIS 4 a IIS 5. È necessario rimuoverla se non si utilizza una rete *Intranet* o se il ***server*** viene connesso al *Web*.

Rimozione di *mapping* di *script* non utilizzati: IIS è stato preconfigurato per il supporto delle estensioni di nomi di *file* comuni, quali asp e shtm. Le richieste di uno di questi tipi di *file* ricevute da IIS vengono gestite da una DLL. Se alcune di queste estensioni o funzionalità non vengono utilizzate, è necessario rimuovere i riferimenti corrispondenti eseguendo la procedura seguente:

- Aprire Gestione *Internet Services*.
- Fare clic con il pulsante destro del *mouse* sul *server Web* e scegliere Proprietà.
- Proprietà principali.
- Scegliere Servizio WWW, quindi Modifica, *HomeDirectory* e infine Configurazione.
- Rimuovere i riferimenti indicati di seguito.

Se non è usato	Disabilitare l'estensione
Reimpostazione di *password* basate su *Web*	.htr, a meno che questa funzionalità non sia assolutamente necessaria!!
Internet Database Connector (in tutti i siti dove è in uso ADO o una tecnologia simile)	.idc
File di inclusione del lato ***server***	.stm, .shtm e .shtml
Stampa *Internet*	.printer
Index Server	.htw, .ida e .idq

Disabilitazione dei percorsi principali: L'opzione Abilita percorsi principali consente di utilizzare .. nelle chiamate a funzioni quali *MapPath*. Per impostazione predefinita, l'opzione è selezionata e deve essere disattivata. A tale scopo, eseguire la procedura seguente:

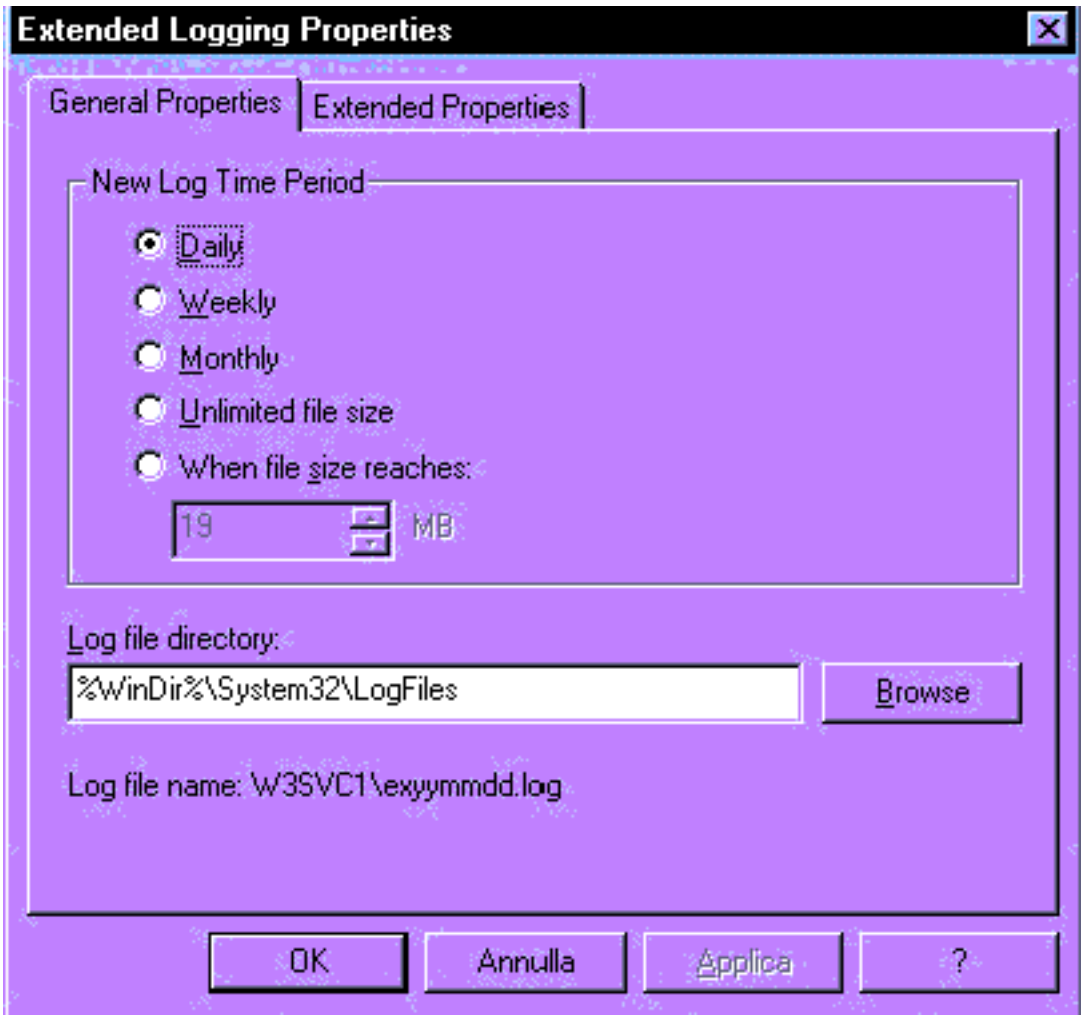
- Fare clic con il pulsante destro del *mouse* sulla radice del sito *Web* e scegliere Proprietà dal menu di scelta rapida.
- Fare clic sulla scheda *Home directory*.
- Fare clic su Configurazione.
- Fare clic sulla scheda Opzioni **applicazione**.
- Deselezionare la casella di controllo Abilita percorsi principali.

Usare e valutare i risultati di uno strumento di memorizzazione delle visite al sito

Per controllare eventuali danni causati da utenti malintenzionati, bisogna tener traccia delle visite al sito. Ciò è realizzato attraverso i *file* di *log*. In questa sezione continuiamo ancora ad occuparci di *Microsoft* IIS.

File di *log* in *Microsoft* IIS

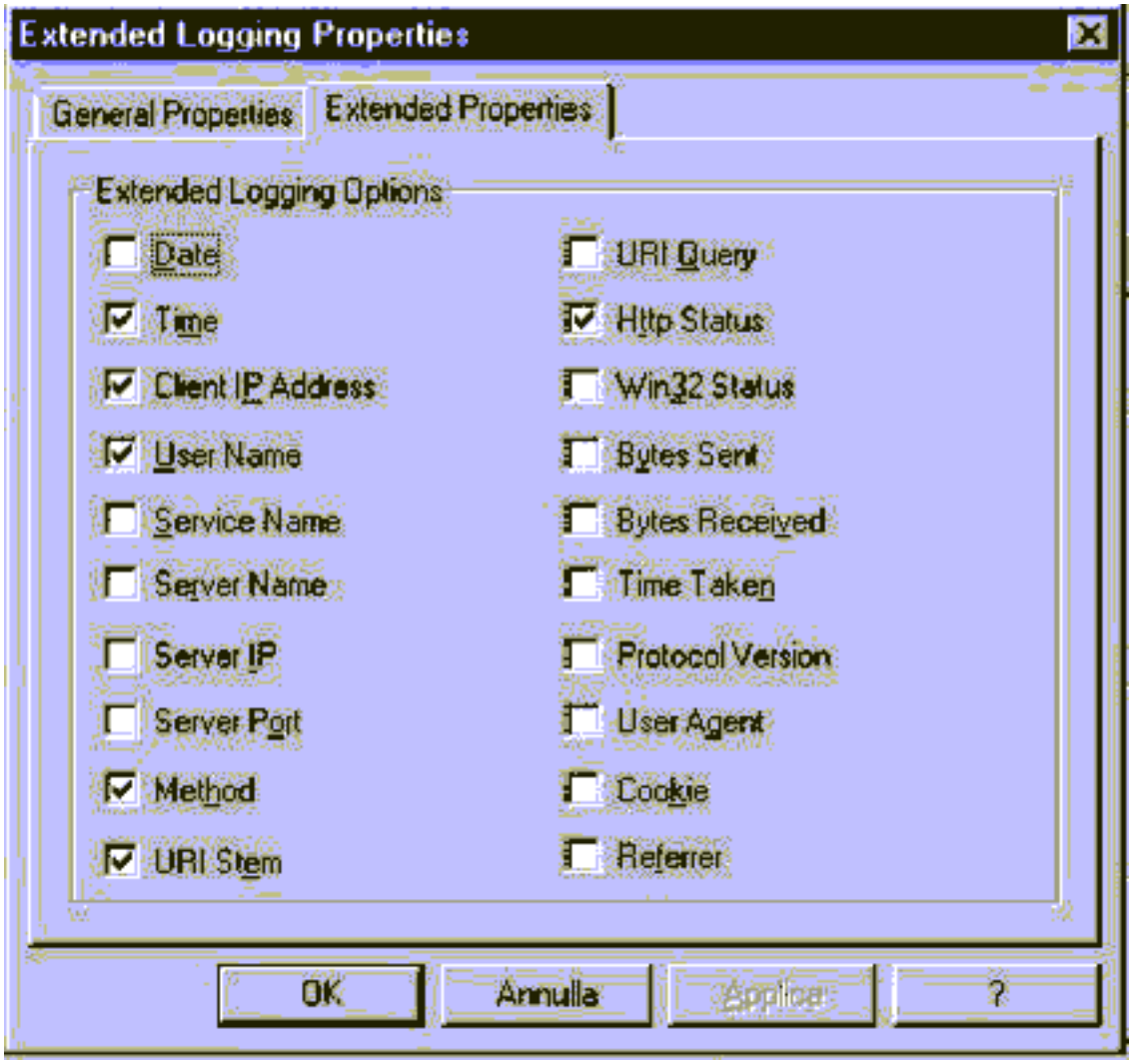
Le operazioni di registrazione relative a un sito *Web* o FTP vengono svolte da moduli che operano indipendentemente dalle altre attività del ***server***. È possibile scegliere il formato dei registri per ogni sito *Web* o FTP. Se si attiva la registrazione a livello di sito, è comunque possibile disattivarla per singole *directory*.



I registri creati da IIS possono essere letti con un *editor* di testo, ma in genere si preferisce caricare i *file* in un programma per la creazione di rapporti. I dati raccolti con la registrazione ODBC vengono registrati in un *database*, da cui è possibile generare rapporti, mentre i registri di conteggio dei processi vengono creati insieme ai normali registri W3C estesi per ogni sito *Web*.

I formati di registrazione utilizzano fusi orari diversi per la registrazione degli orari. Il formato di registrazione W3C esteso utilizza l'orario UTC (*Universal Time Coordinate*), precedentemente noto come GMT (*Greenwich Mean Time*). Gli altri formati utilizzano orari locali. Gli orari riportati nei *file* registro indicano il tempo impiegato dal ***server*** per elaborare richieste e risposte, escluso il tempo impiegato dai dati per arrivare al *client* e il tempo impiegato dal *client* per elaborare i dati.

Il *folder Extended Properties*, permette di scegliere gli obiettivi che dovranno essere monitorati dal LOG. La grandezza dei *file* di *Log* sarà proporzionale al numero di opzioni selezionate. È possibile indicare nel *file* di *Log* i campi che vorremmo vedere. Questo argomento sarà trattato diffusamente in seguito.



La parte relativa alla generazione dei LOG, è molto importante da valutare. Infatti la mancata osservanza di alcune regole fondamentali potrebbe portare a creare dei *file* di LOG da qualche *Gigabyte*. Abilitando il LOG (ricordiamo che per *default* è abilitato), ogni richiesta effettuata al ***server*** (anche quelle che non hanno esito e quelle errate) verranno registrate. Dal più semplice LOG di testo, al formato *Microsoft*, fino a W3C che è un formato esteso di *Logging* che permette la personalizzazione degli obbiettivi da monitorare. Il LOG però assorbe risorse dal ***server***, è quindi consigliata l'attivazione solo quando se ne ritiene fondamentale il suo uso. Nelle proprietà del *Logging* è possibile stabilire se deve essere: Giornaliero Settimanale Mensile Illimitato (Sconsigliatissimo) personalizzato nella grandezza. È possibile stabilire in quale *directory* dovrà essere contenuto (*default* WINNT/SYSTEM32/LogFiles).

Formati dei file registro

È possibile scegliere il formato utilizzato dal *server Web* per registrare l'attività dell'utente. Sono disponibili i seguenti formati:

- Formato Registrazione W3C estesa.
- Formato Registrazione *Microsoft* IIS.
- Formato Registrazione comune NCSA.
- Registrazione ODBC.

I formati Registrazione W3C estesa, *Microsoft* IIS e NCSA sono formati di testo ASCII. I formati Registrazione W3C estesa e NCSA registrano le informazioni sugli anni in un formato a quattro cifre, mentre il formato *Microsoft* IIS utilizza per gli anni un formato a due cifre ed è compatibile con le versioni precedenti di IIS. È inoltre possibile creare formati di registrazione personalizzati che prevedano esclusivamente i campi di interesse.

Formato Registrazione W3C estesa

Il formato Registrazione W3C estesa è un formato ASCII personalizzabile che include numerosi campi. È possibile scegliere i campi di interesse e omettere quelli non desiderati, così da limitare le dimensioni del registro. I campi sono separati da spazi. Gli orari vengono registrati in formato UTC (*Universal Time Coordinate*). Per informazioni sulla personalizzazione di questo formato, vedere Personalizzazione della registrazione W3C estesa. Per ulteriori informazioni sulla specifica alla base della registrazione W3C estesa, vedere il sito di W3C all'indirizzo http://www.w3.org (informazioni in lingua inglese).

Nell'esempio che segue vengono riportate alcune righe da un *file* per il quale sono stati scelti i campi: Ora, Indirizzo IP *client*, Metodo, Origine URI, Stato HTTP e Versione HTTP.

#Software: Microsoft Internet Information Services 5.0
#Version: 1.0 #Date: 1998-05-02 17:42:15
#Fields: time c-ip cs-method cs-uri-stem sc-status cs-version 17.42.15
172.16.255.255 GET /default.htm 200 HTTP/1.0

Queste voci indicano che il 2 maggio 1998 alle ore 17.42 (ora UTC) un utente con HTTP versione 1.0 e indirizzo IP 172.16.255.255 ha eseguito un comando HTTP *GET*

per il *file Default.htm*. La richiesta è stata soddisfatta senza errori. Il campo *#Date:* indica quando è avvenuta la registrazione della prima voce, ovvero quando è stato creato il registro. Il campo *#Version:* indica che è stato utilizzato il formato di registrazione W3C.

È possibile selezionare qualsiasi campo, ma alcuni potrebbero non contenere informazioni per alcune richieste. Nei campi selezionati che non presentano informazioni registrabili viene visualizzato un trattino (-).

Formato Registrazione Microsoft IIS

Il formato di registrazione *Microsoft* IIS è un formato ASCII fisso, non personalizzabile, in grado di registrare più informazioni rispetto al formato Registrazione comune NCSA. Include elementi di base, quali l'indirizzo IP e il nome dell'utente, la data e l'ora della richiesta, il codice di stato HTTP e il numero di *byte* ricevuti, nonché elementi dettagliati, quali il tempo trascorso, il numero di *byte* inviati, l'azione, ad esempio un processo di scaricamento eseguito tramite un comando *GET*, e il *file* di destinazione. I vari elementi sono separati da virgole, e questo semplifica la lettura di tale formato rispetto agli altri formati ASCII che utilizzano gli spazi come separatori. L'ora viene registrata in base al fuso orario locale.

Quando si apre un *file* registro in formato *Microsoft* IIS con un *editor* di testo, le voci in esso contenute risulteranno simili a quelle riportate negli esempi seguenti:

> 192.168.114.201,-, 20/03/98, 7.55.20, W3SVC2, VENDITE1, 192.168.114.201, 4502, 163, 3223, 200, 0, GET, LogoDip.gif 172.16.255.255, anonymous, 20/03/98, 23.58.11, MSFTPSVC, VENDITE1, 192.168.114.201, 60, 275, 0, 0, 0, PASS, intro.htm

Le voci dell'esempio precedente vengono spiegate nelle tabelle riportate di seguito. La prima riga di entrambe le tabelle deriva dalla seconda istanza del sito *Web* (che appare nella sezione Servizio come W3SVC2). L'ultima riga deriva dalla prima istanza del sito FTP (che appare nella sezione Servizio come MSFTPSVC1). Per motivi legati alla larghezza della pagina, è stato necessario utilizzare due tabelle per descrivere l'esempio.

Indirizzo IP dell'utente	Nome utente	Data	Ora	Servizio e istanza	Nome del *computer*	Indirizzo IP del *server*
192.168.114.201	-	03/20/98	7:55:20	W3SVC2	VENDITE1	172.21.13.45
172.16.255.255	anonimo	03/20/98	23:58:11	MSFTPSVC1	VENDITE1	172.21.13.45

Tempo	*Byte* ricevuti	*Byte* inviati	Codice dello stato del servizio	Codice dello stato di *Win* 2000	Tipo di richiesta	Destinazione dell'operazione
4502	163	3223	200	0	*GET*	LogoDip.gif
60	275	0	0	0	[376] PASS	intro

Nell'esempio precedente, la prima voce indica che alle ore 7.55 del giorno 20 marzo 1998 un utente anonimo con indirizzo IP 102.168.114.201 ha eseguito un comando HTTP *GET* per scaricare il *file* immagine LogoDip.gif da un ***server*** chiamato VENDITE1 con indirizzo IP 172.21.13.45. Il tempo di elaborazione necessario per completare questa richiesta HTTP di 163 *byte* è stato di 4502 millisecondi, ovvero 4,5

secondi. All'utente anonimo sono stati restituiti 3223 *byte* di dati, senza errori. Tutti i campi presenti nel *file* registro terminano con una virgola (,). Eventuali trattini vengono utilizzati come segnaposto per indicare la non disponibilità di un valore valido per il campo.

Formato Registrazione comune NCSA

Il formato Registrazione comune NCSA è un formato ASCII fisso, non personalizzabile, disponibile per i siti *Web*, ma non per i siti FTP. Registra le informazioni di base relative alle richieste degli utenti, ad esempio il nome dell'*host* remoto, il nome utente, la data, l'ora e il tipo di richiesta, il codice di stato HTTP e il numero di *byte* ricevuti dal ***server***. Le voci sono separate da spazi e l'ora viene registrata in base al fuso orario locale. Quando si apre un *file* registro in formato comune NCSA con un *editor* di testo, le voci in esso contenute risulteranno simili a quelle riportate nell'esempio seguente:

172.21.13.45- ROMA\pippo [08/Apr/1998.17.39.04 -0800] GET /script/iisadmin/ism.dll?http/serv HTTP/1.0 200 3401

Nella voce precedente, il secondo campo, che dovrebbe contenere il nome del registro remoto dell'utente, è vuoto ed è rappresentato dal trattino che segue l'indirizzo IP 172.21.13.45. Questa voce viene interpretata nelle tabelle seguenti. Per motivi legati alla larghezza della pagina, è stato necessario utilizzare due tabelle per illustrare l'esempio.

Nome dell'*host* remoto	Nome utente	Data	Ora e differenza GMT
172.21.13.45	ROMA\sergio	08/Apr/1998	17:39:10 -0800

Tipo di richiesta	Stato del servizio	*Byte* inviati
GET /script/iisadmin/ism.dll?http/serv HTTP/1.0	200	3401

La voce precedente indica che alle ore 17.39 dell'8 aprile 1998 l'utente Sergio appartenente al dominio ROMA, con indirizzo IP 172.21.13.45, ha eseguito un comando HTTP *GET* per scaricare un *file*. All'utente sono stati restituiti, senza errori, 3401 *byte* di dati.

Formato Registrazione ODBC

Il formato di registrazione ODBC registra una serie di campi di dati fissi in un *database* ODBC. L'ora viene registrata in base al fuso orario locale. Se si sceglie questo formato di registrazione è necessario specificare e configurare il *database* in cui registrare i dati. Per utilizzare la registrazione ODBC è necessario procedere come indicato di seguito:

- Creare un *database* contenente una tabella con i campi necessari per la registrazione delle informazioni. In IIS è disponibile un *file* modello SQL che può essere eseguito in un *database* SQL per creare una tabella predisposta per la registrazione dei dati di IIS. Il *file* si chiama Logtemp.sql, nella *directory* \IISRoot. Se durante l'installazione sono state confermate le impostazioni predefinite, la *directory* \IISRoot è una sotto*directory* di \WindowsNT\System32. I campi obbligatori sono:

Nome campo	Tipo campo
ClientHost	*Varchar*(255)
Username	*Varchar*(255)
LogTime	*datetime*
Service	*Varchar*(255)
Machine	*Varchar*(255)
ServerIP	*Varchar*(50)
ProcessingTime	*int*
BytesRecvd	*int*
BytesSent	*int*
ServiceStatus	*int*
Win32Status	*int*
Operation	*Varchar*(255)
Target	*Varchar*(255)
Parameters	*Varchar*(255)

- Assegnare al *database* un nome DSN (*Data Source Name*), che verrà utilizzato dal *software* ODBC per trovare il *database*.
- In IIS specificare il nome del *database* e della tabella.

Conteggio dei processi e nomi dei file registro

Conteggio dei processi: È una nuova funzione di IIS e aggiunge campi al *file* registro in formato W3C esteso allo scopo di registrare informazioni sull'utilizzo delle risorse della *CPU* del ***server*** da parte dei siti *Web*. Queste informazioni vengono quindi utilizzate per stabilire se i siti impiegano eccessive risorse della *CPU* o per rilevare *script* o processi *CGI* che non funzionano in modo corretto. Il conteggio dei processi può essere attivato a livello di sito. Non fornisce dettagli sull'utilizzo della *CPU* da parte delle singole applicazioni; infatti, registra le informazioni relative solo alle applicazioni *out-of-process*. È disponibile solo per i siti *Web* e viene registrato solo se è selezionato il formato di registrazione W3C estesa. I dati relativi al conteggio dei processi vengono registrati nel *file* insieme agli altri dati. Le informazioni raccolte durante il conteggio dei processi possono essere utilizzate per decidere se attivare o meno la limitazione dei processi in un sito *Web*, ovvero se definire limiti per il tempo del processore utilizzato da un sito.

Nomi dei *file* registro: Le prime lettere dei nomi dei *file* registro indicano il formato di registrazione mentre i restanti numeri indicano la sequenza o l'intervallo temporale di creazione dei *file*. Nella tabella riportata di seguito vengono fornite informazioni più dettagliate. Le lettere in corsivo rappresentano cifre: *nn* indica cifre sequenziali, mentre *aa* indica l'anno, *mm* il mese, *ss* la settimana del mese, *gg* il giorno e *hh* l'ora nel formato 24 ore.

Formato	Criterio di creazione dei nuovi *file* registro	Modello del nome di *file*
Registrazione *Microsoft* IIS	In base alle dimensioni del *file*	**inetsv***nn***.log**
	Ogni ora	**in***aammgghh***.log**
	Ogni giorno	**in***aammgg***.log**
	Ogni settimana	**in***aammss***.log**
	Ogni mese	**in***aamm***.log**
Registrazione comune NCSA	In base alle dimensioni del *file*	**ncsa***nn***.log**
	Ogni ora	**nc***aammgghh***.log**
	Ogni giorno	**nc***aammgg***.log**
	Ogni settimana	**nc***aammss***.log**
	Ogni mese	**nc***aamm***.log**
Registrazione W3C estesa	In base alle dimensioni del *file*	**extend***nn***.log**
	Ogni ora	**ex***aammgghh***.log**
	Ogni giorno	**ex***aammgg***.log**
	Ogni settimana	**ex***aammss***.log**
	Ogni mese	**ex***aamm***.log**

Approfondimento

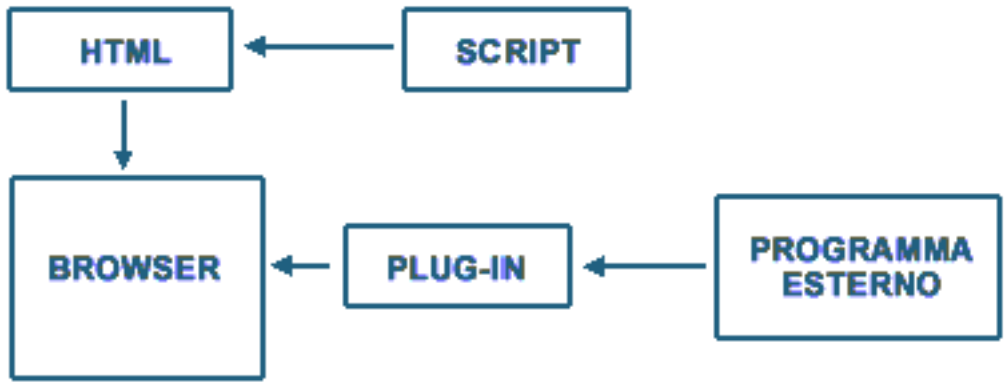
L'elaborazione dei dati su client

Cosimo Laneve

16.2.1 (Confrontare gli strumenti attualmente disponibili di gestione di un sito Web).

Linguaggi di script

Attualmente si tende a scaricare sul *computer* dell'utente piccoli programmi (*Javascript* o ***applet Java***) che svolgano parte dell'elaborazione in locale. Il vantaggio è sia nel tempo di risposta (più veloce perché si attendono meno dati dal ***server***) sia nel risparmio di elaborazione centrale, sia di altro tipo (per esempio può essere utile per criptare dati in modo più sicuro o per altro ancora). In sé, quindi, non è una cattiva scelta, però è più pericolosa, perché si basa sull'attrezzatura *hardware* e *software* dell'utente, che è **variabile**.



Architettura client con script e plug-in

Per rendere dinamica una pagina *Web*, si utilizzano i cosiddetti **linguaggi di *scripting***, che sono dei veri e propri linguaggi di programmazione. Si distinguono dagli altri comuni, e più anziani, linguaggi perché non sono compilati ma **interpretati**. A seconda dell'**interprete** si parla di *scripting*:

- **server-side**: l'esecuzione dello *script* avviene sul ***server*** e propaga il risultato sul *client* che ne ha chiesto l'elaborazione;
- **client-side**: l'elaborazione avviene all'interno del *client* che ha caricato il documento html.

Se è il *server-Web (Apache, Internet Information Server)* che elabora uno *script* si parla di *scripting server-side (SSI, server side include)*, viceversa, se è il *browser (Netscape, Internet Explorer)* si parla di *scripting client-side*. Il linguaggio lato *client* per eccellenza, ma anche per anzianità, è sicuramente **JavaScript**.

Quando le operazioni da compiere non sono troppo complesse o troppo pesanti per il *browser*, si adottano gli *script client-side*, per esempio, quando occorre validare il contenuto di un campo di testo o realizzare piccole funzioni che sfruttano le informazioni del *browser*.

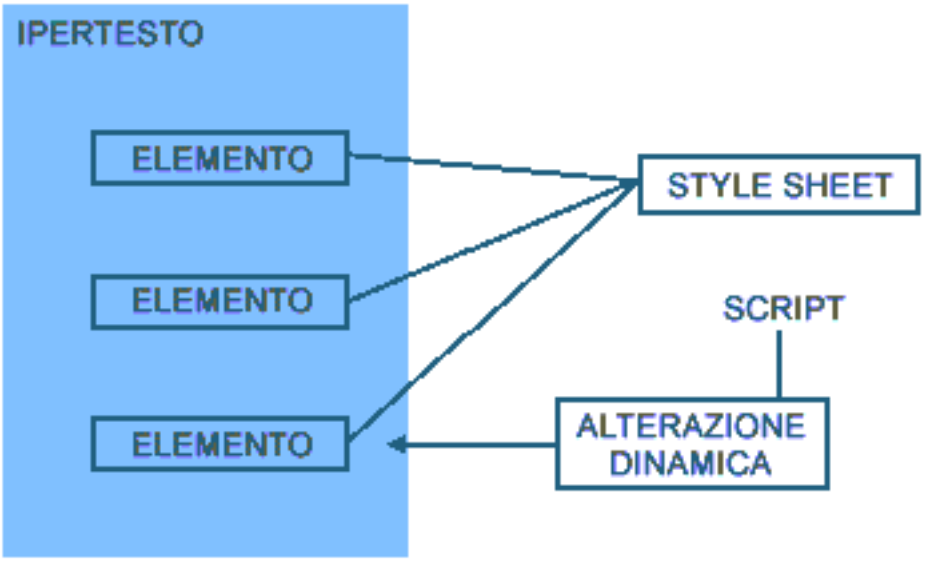
La necessità di uno scambio di informazioni più corposo tra il *client* e il ***server*** (quando, per esempio, si consulta una base dati) richiede l'utilizzo di *script server-side*. Lato ***server*** ci sono due modi per risolvere questa esigenza. Il primo è quello di realizzare *script* ***CGI***, programmi scritti in qualsiasi linguaggio (*C, C++, Visual Basic*) che eseguono delle operazioni e inviano il risultato di queste sotto forma html al *browser*. Il secondo, è quello di utilizzare linguaggi di *scripting* il cui codice è inserito direttamente nella pagine e processato dal ***server*** prima di essere inviato al *client* (ASP, PHP).

Abbiamo così chiarito sommariamente la differenza tra *scripting* ***server-side*** e **client-side**.

È necessario sottolineare l'importanza di ottenere precise garanzie riguardo alla compatibilità di questi pezzetti di *software* che si scaricano sul *computer* dell'utente. Devono funzionare bene sulle diverse piattaforme (PC, *Mac*, e *Unix*) e con diversi *browser* (*Explorer* e *Netscape* delle *release* più diffuse). *Macintosh* e *Netscape* solitamente vengono trascurati perché costituiscono una minoranza, che tuttavia non è trascurabile, soprattutto presso i consumatori e se il sito è destinato a grandi numeri.

Cosa è DHTML

DHTML o *Dynamic* HTML è un'evoluzione di HTML, nato per rendere meno statico il codice HTML delle pagine *standard*: purtroppo allo stato attuale il suo uso è implementato in maniera differente dalle versioni 4.0 dei *Web browser* quali *Netscape Navigator*, MS *Internet Explorer* ed altri, per cui non esiste uno *standard* unico per lo sviluppatore che desideri scrivere codice *cross-browser* ovvero non specifico per un *browser* particolare, come sarebbe invece utile.



Documento DHTML

HTML dinamico permette, attraverso l'uso congiunto di CSS (*Cascading Style Sheet*), *script* e codice HTML, di creare pagine *Web* interattive mediante i cosiddetti stili dinamici. Nella pratica ciò significa che gli sviluppatori di pagine *Web* hanno, grazie a queste nuove specifiche, la possibilità di modificare l'aspetto di un documento senza che questo venga ricaricato.

Un uso particolarmente interessante degli stili dinamici è quello che consente la creazione di sommari espandibili che, a seconda delle scelte effettuate, viene esposto o nascosto alla vista del *browser*. Uno degli esempi più evidenti di tale struttura è visibile nella *home page* di *Microsoft*.

Il modello di stile dinamico considerato negli esempi di questo approfondimento è compatibile esclusivamente con MS *Internet Explorer* 4, mentre *Netscape* ha percorso una strada diversa adottando il modello JASS (*JavaScript Accessible Style Sheets*). Tale incompatibilità è il freno maggiore allo sviluppo di **DHTML** presso i creatori di pagine *Web*, che poco tollerano l'idea di creare siti dinamici che discriminino l'accesso agli utenti dell'uno o dell'altro *browser*.

Stili dinamici

Gli stili dinamici permettono agli sviluppatori di rendere inizialmente visibile solo una parte del documento, con la possibilità, attraverso la definizione di elementi e classi in **DHTML**, di espanderne la struttura e visualizzarne il contenuto in modo analitico. Il codice che segue è un esempio molto semplice di tale struttura:

```
<HTML>
<HEAD>
<TITLE>Testo espandibile</TITLE>
<STYLE TYPE=text/css>
body {background:white}
.testo {color:#000080; font-size:10pt; FONT-FAMILY: verdana; cursor:help}
.times {color:red; font-size:14pt; FONT-FAMILY: times new roman;}
.vuoto {display:none}
</STYLE>
<SCRIPT LANGUAGE=JavaScript>
function stile(st) {
menu.className=st
}
</SCRIPT>
</HEAD>
<BODY ONCLICK=outliner();>
<H1 CLASS=fisso child=menu> DHTML </H1>
Questo testo viene formattato automaticamente senza dover ricaricare la
pagina.
<p> </p>
<p><input type=radio value=V1 checked name=R1 onclick=stile(times)>times
14 pt<br>
<input type=radio value=V2 name=R1 onclick=stile(testo)>verdana 10 pt<br>
<input type=radio value=V3 name=R1 onclick=stile(vuoto)>nascondi il testo</p>
<p> </p>
<DIV ID=menu CLASS=times>
testo formattato...
</DIV>
```

In questo esempio vengono create tre classi: fisso, espandibile e vuoto. La classe fisso stabilisce gli elementi che, una volta cliccati, cambiano aspetto alla struttura e si

associa all'attributo *child*. Quest'ultimo contiene l'ID dei dati da nascondere o visualizzare, e identifica, in modo univoco, gli elementi di questo genere all'interno della struttura. Nel caso specifico la classe fisso determina, oltre al colore blu del testo, il tipo di puntatore che il *mouse* deve assumere quando si trova su un elemento della classe stessa. Quando si clicca sull'elemento fisso, il nome di classe dei dati cambia da vuoto a espandibile, o viceversa, generando l'effetto desiderato.

Gli *SCRIPT* indicano al *browser* come interpretare le informazioni presenti in una pagina *Web*. In questo modo, relativamente alla marcatura presente tra <SCRIPT> </SCRIPT>, il *browser* delega all'autore la gestione degli eventi del documento. Nel momento in cui il *browser* legge, attraverso una particolare procedura chiamata di *parser*, la marcatura del documento ed incontra <SCRIPT>, ne passa il contenuto all'elaboratore di *script*, per poi continuare l'interpretazione del resto della pagina.

I *browser* che non supportano la gestione di *script* ignorano quanto posto all'interno di <SCRIPT></SCRIPT>.

Plug-in

Esistono attualmente sul mercato diverse soluzioni di distribuzione tramite *browser* dei dati *host* ai *desktop* che si basano sulla tecnologia di *Internet*. Tali soluzioni utilizzano vari metodi: tra i quali ricordiamo le ***applet***, ***ActiveX*** di *Microsoft* e la pubblicazione *host*. In questo documento verrà fornita una descrizione di ognuna di queste soluzioni e dei relativi punti di forza e limiti.

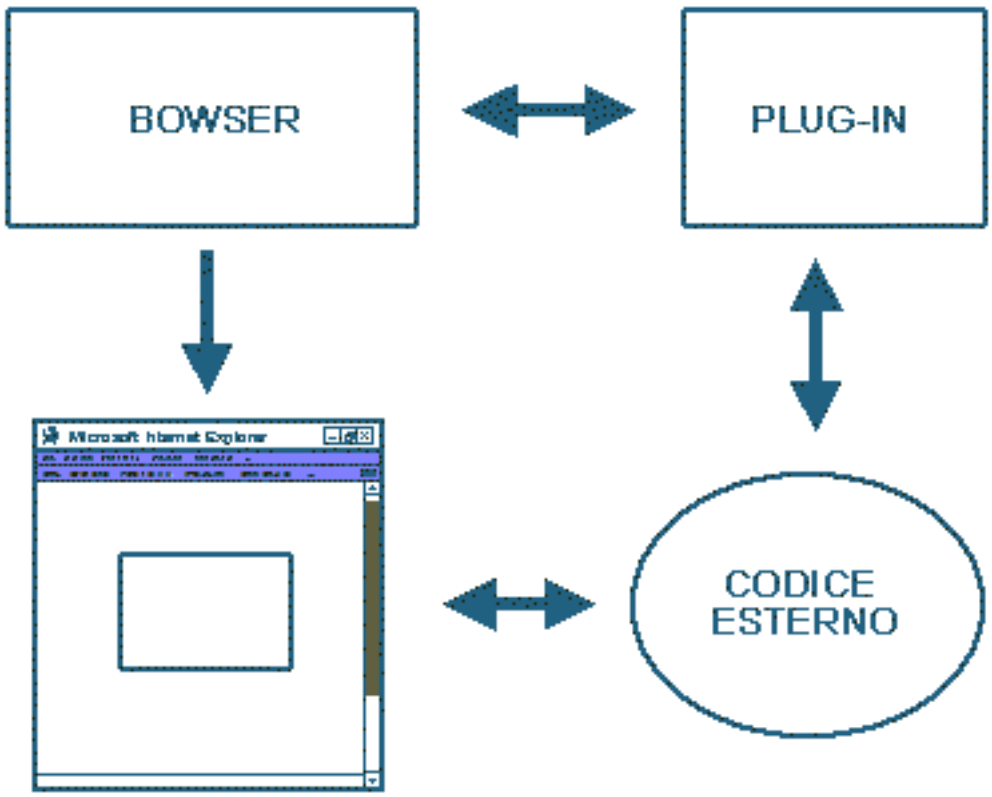
Applet Java e *ActiveX*

Le ***applet*** scaricabili si avvalgono delle nuove tecnologie distribuite di elaborazione dei dati, come ***Java*** e ***ActiveX***, per implementare applicazioni di connettività pienamente funzionali che possono essere scaricate direttamente nel *desktop*. Queste ***applet*** si integrano totalmente nell'ambiente del *browser* e in genere possono essere automatizzate con strumenti di *scripting* per *client* tipo *VB Script* e *Java Script*. È possibile sviluppare ***applet*** con livelli di funzionalità paragonabili a quelli dei prodotti di connettività *desktop* tradizionali, ma con un inconveniente: le ***applet*** sono sì sempre più funzionali, ma allo stesso tempo diventano più grandi e meno facili da scaricare. La maggior parte delle ***applet*** disponibili sul mercato offre meno funzionalità rispetto ai tradizionali prodotti *desktop*. È però probabile che, una volta acquisita maggiore esperienza con ***Java*** e ***ActiveX*** e trovata una soluzione circa i limiti della larghezza di banda delle reti, le ***applet*** raggiungano il livello di piena funzionalità dei *client* attualmente disponibili.

La scelta dell'***applet*** ideale per un ambiente specifico dipende in gran parte dalla tecnologia sulla quale tale ***applet*** si basa: ***Java*** o ***ActiveX***. Attualmente esistono differenze significative tra le due tecnologie, anche se nel tempo è probabile che tali differenze diminuiscano permettendo di scegliere tra funzionalità diverse piuttosto che tra piattaforme di sviluppo diverse. In generale, i fattori che differenziano ***Java*** e ***ActiveX*** dipendono dall'approccio delle due tecnologie ad aspetti quali protezione, persistenza e supporto per piattaforme multiple. Per esempio adesso potreste non vedere l'activeX perché non è consentito dal vostro *browser*, oppure, se per *default* fosse consentito, questo potrebbe creare problemi di sicurezza.

Protezione

Le ***applet Java*** sono molto sicure. Con il linguaggio *Java* ogni ***applet*** scaricabile viene eseguita all'interno della propria macchina virtuale e non può uscire dai propri confini, né può leggere o scrivere sul disco rigido locale o in posizioni locali di memoria. Chiaramente, questo pone anche limiti alla funzionalità delle ***applet Java***. Con ***ActiveX***, invece, le ***applet*** hanno accesso completo alla memoria della macchina *client* e al supporto di memorizzazione locale. Sebbene utile, questa caratteristica rende gli oggetti ***ActiveX*** potenzialmente poco sicuri. Per risolvere il problema, è possibile registrare e assegnare una firma digitale alle ***applet ActiveX*** e garantire quindi l'origine dell'***applet***. Nel caso delle ***applet*** per la connettività, in genere acquistate presso un fornitore di fiducia e richiamate da una *Intranet* aziendale protetta, la sicurezza può passare in secondo piano.



Interazioni tra browser, plug-in e codice esterno (applet)

Persistenza

Gli oggetti ***ActiveX*** sono persistenti, cioé rimangono all'interno della macchina locale dopo essere stati scaricati, mentre le ***applet Java*** vengono caricate in memoria e quindi eliminate quando il *browser* viene scaricato. La persistenza offre il vantaggio di non dover scaricare l'***applet*** ogni volta che deve essere utilizzata, ovvero permette di

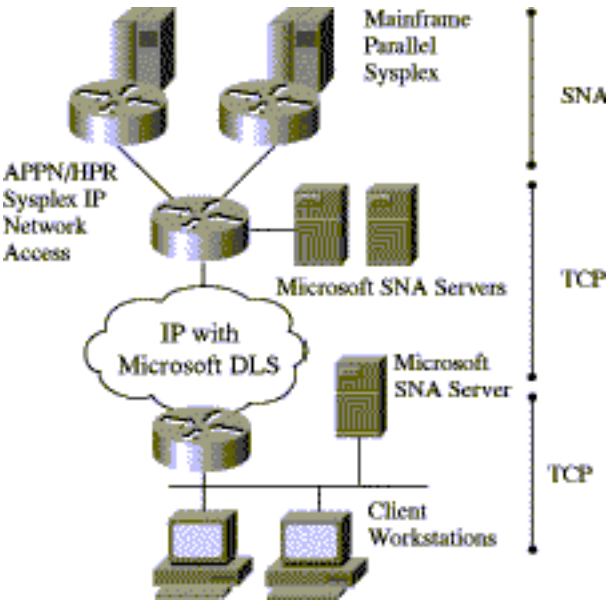
conservare una versione locale dell'***applet*** riducendo notevolmente il traffico di rete che il caricamento invece comporta. L'oggetto ***ActiveX*** viene scaricato solo nel caso in cui la versione presente sul ***server*** risulti più recente della versione disponibile sul *client*. Questa differenza è fondamentale a livello di rete, se accompagnata dall'uso di applicazioni specifiche eseguite sui *client* di piccole dimensioni, come le ***applet*** per la connettività.

Supporto per piattaforme multiple

Al momento, le ***applet Java*** offrono un maggiore supporto per piattaforme multiple rispetto agli oggetti ***ActiveX***. I controlli ***ActiveX*** funzionano solo sulle piattaforme a 32 *bit* di *Windows* e su *Mac OS*. Lo sviluppo del supporto per altre piattaforme è stato affidato a una società di terze parti. ***Java*** supporta non solo *Windows* 95 e NT, ma anche *Windows* a 16 *bit*, DOS, *Macintosh* e praticamente ogni variante di *UNIX*. Il supporto per piattaforme multiple diventa un fattore fondamentale al momento della scelta di un'***applet*** per la connettività solo se l'utente è in presenza di un ambiente misto con più piattaforme.

Pubblicazioni *Web-to-host*

I prodotti per la pubblicazione *host* sono strumenti di sviluppo *middleware* che consentono ai programmatori aziendali di creare applicazioni basate sul ***server*** in grado di incorporare i dati *host* in documenti HTML. Questo approccio è diverso dalla traduzione diretta precedentemente descritta, in quanto in questo caso lo sviluppatore mantiene il controllo sui dati *host* da visualizzare. Gli sviluppatori sono dunque in grado di riprogrammare il flusso dei *task* dell'**applicazione** *host*, fornendo agli utenti finali un'**applicazione** basata sul *Web* più intuitiva e semplice da usare.



Architettura di una rete di server Microsoft

I dati possono essere integrati con qualsiasi altro contenuto di tipo HTML, come la grafica, i collegamenti ipertestuali e il contenuto interattivo. Inoltre, i dati *host* provenienti da più fonti possono essere combinati con altri dati in modo da formare un

unico documento HTML. Esistono due tipi di dati che vengono in genere pubblicati dagli sviluppatori in HTML: i dati memorizzati nei *database host* e i dati che vengono visualizzati sullo schermo da un'**applicazione** basata su *host*. Molte soluzioni garantiscono l'accesso a un solo tipo di dati, mentre molti progetti di pubblicazione *host* richiedono una combinazione di dati provenienti da entrambe le fonti.

Una grande percentuale dei dati aziendali risiede in grandi *database host* come DB2. Le soluzioni per la pubblicazione *host* che garantiscono l'accesso a questo tipo di dati utilizzano in genere un *driver* del *database*, spesso basato su ODBC, che traduce le interrogazioni del *database* in comandi interpretabili dal *database host*. I risultati dell'interrogazione possono quindi essere pubblicati in formato HTML.

È comunque possibile accedere alla maggior parte dei dati *host* solo attraverso le applicazioni *host* che visualizzano i dati sullo schermo di un terminale. I prodotti per la pubblicazione *host* che danno accesso a dati di questo tipo devono offrire un insieme di strumenti di programmazione in grado di semplificare l'interazione del programma con l'*host*. L'operazione più difficile nell'estrazione dei dati basati sullo schermo consiste nel creare il codice necessario per controllare la navigazione tra le varie schermate dell'**applicazione** *host*, fino a raggiungere la schermata contenente i dati.

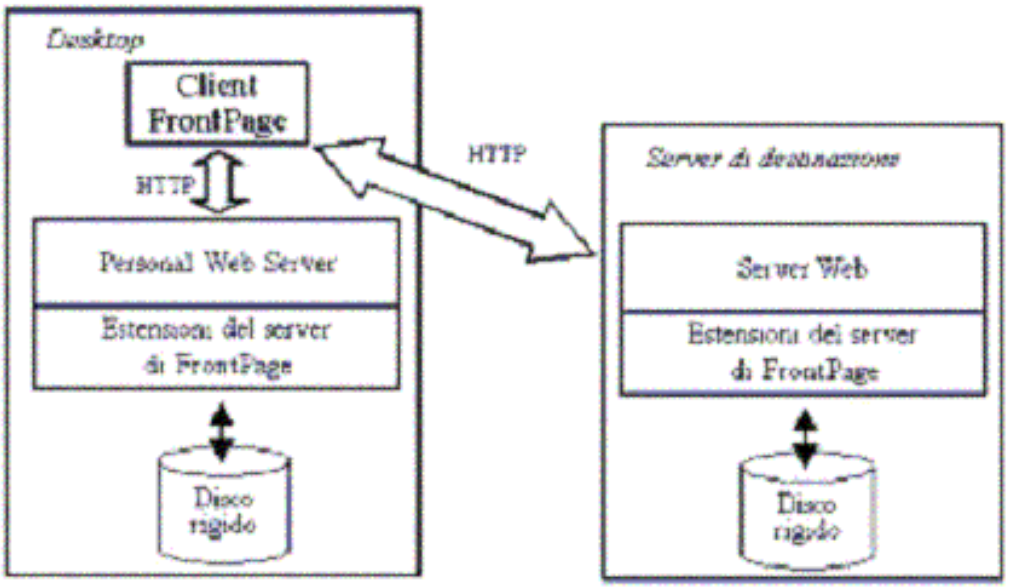
JavaScript

Cosimo Laneve

16.2.1 (Confrontare gli strumenti attualmente disponibili di gestione di un sito Web).

JavaScript

JavaScript è un **linguaggio di *scripting***, ed sicuramente il più usato. Gli *script* realizzati tramite questo linguaggio possono essere incapsulati nel codice HTML. Tramite *JavaScript* è possibile rispondere alle azioni dell'utente. Ad esempio si possono convalidare i *form* prima che questi vengano trasmessi al ***server*** per una elaborazione magari errata. Infatti quando un utente inserisce un dato in un *form* questo deve essere inviato al ***server***, che dopo averlo elaborato spedisce una risposta. Grazie a *JavaScript* il *form* può essere inviato solo dopo che sia stato controllato.



Interazioni di Javascript con sistema operativo, form, eventi e funzioni

Ma la potenza di *JavaScript* non si limita solo ai *form*: con esso si possono effettuare tantissimi tipi di *script* che si possono occupare dalla grafica alla *utility* più disparate. *JavaScript* è un linguaggio interpretato, infatti questo viene inviato al *client* in *file* ASCII, e quindi in chiaro, che vengono elaborati riga per riga nei *browser* in modalità *runtime*.

- **Pro.** Il **linguaggio di *scripting*** è più sicuro e affidabile perché in chiaro. Il codice *JavaScript* viene eseguito sul *client*, per cui i ***server*** sono sollecitati molto di meno. Codice visibile e leggibile da chiunque, ho inserito questa voce nei pro perché penso che solamente condividendo le risorse si possa avanzare tecnologicamente.
- **Contro.** Occorre scaricare completamente il codice dello *script* per essere eseguito. Con *JavaScript* non è possibile fare tutto, questo per ragioni di sicurezza, per cui per effettuare alcune operazioni occorre ricorrere a linguaggi più potenti tipo ***Java*** (*JavaScript* e ***Java*** non sono affatto la stessa cosa).

Le versioni

Nel corso degli anni la *NetScape Communications Corporation* ha dato vita a sempre più aggiornate versioni di *JavaScript*. La prima versione standardizzata di *JavaScript* fu riconosciuta nel giugno dell'1997 dall'ECMA, era la versione 1.1 e fu definita *ECMaScript* o *ECMA-262*. Per dovere di cronaca l'*ECMA* è una associazione internazionale di industrie basate sull'Europa dedicata alla standardizzazione di sistemi di comunicazioni e informazioni.

A *JavaScript* della *NetScape*, disponibile dalla release 2.0, rispose la *Microsoft* con *JScript* disponibile su *Internet Explorer* 3.0 simile a *JavaScript*.

	JavaScript				***JScript***		
Versioni	1.0	1.1	1.2	1.3	1.0	3.0	5.0
NS 2.0	*						
NS 3.0	*	*					
NS 4.0	*	*	*				
NS 4.06	*	*	*	*			
MSIE 3.0	*				*		
MSIE 4.0	*	*	*		*	*	
MSIE 5.0	*	*	*	*	*	*	*

NS sta per *NetSCape*, MSIE per *Internet Explorer*.

Tag Script e NoScript

Per inserire uno *script* all'interno di una pagina HTML occorre utilizzare il ***tag*** <SCRIPT>. Questo ***tag*** è possibile inserirlo in qualsiasi posizione della pagina, l'importante è chiuderlo.

Gli *script* possono essere posizionati tra i ***tag*** <HEAD> in modo che siano caricati per primi, importante se si utilizzano delle variabili per gestire la pagina e per inserirvi delle funzioni che vengono avviati da eventi attivati sulla pagina, oppure in qualsiasi parte della pagina.

Se ne possono inserire in una misura qualsiasi l'importante è chiuderli. Il *browser*, infatti, legge la pagina dall'alto verso il basso, quando incontra il *Tag* <SCRIPT> continua a leggere sempre nello stesso verso ma interpreta le righe in maniera diversa, per cui se il ***tag*** non viene chiuso con l'apposito ***tag*** </SCRIPT> anche la restante parte della pagina viene interpretata come codice *JavaScript* con conseguente errore nella esecuzione.

In caso di errore nella fase di esecuzione si possono verificare due comportamenti diversi da parte del *browser*:

- viene visualizzata la pagina ma il codice errato non viene eseguito.
- Se lo *script* genera un *loop* la pagina può o restare bianca o essere parzialmente visualizzata perché l'esecuzione del codice HTML è stato interrotto, e quindi verrà visualizzato solo il codice antecedente allo *script* che ha generato il *loop*.

Per inserire uno *script* in una pagina occorrono queste righe:

```
<SCRIPT>
<!-- Istruzioni JavaScript --!>
</SCRIPT>
```
Dato che i linguaggi di *scripting* sono diversi occorre specificare a quale linguaggio associare lo *script*, e quindi evitare che si utilizzi quello non voluto:

```
<SCRIPT language=JavaScript>
<!-- Istruzioni JavaScript --!>
</SCRIPT>
```
In questo modo si indica che lo *script* è in codice *JavaScript*. Se l'utente utilizza un *browser* che non supporta *JavaScript*, oppure è disabilitato, esiste un ***tag*** grazie al quale è possibile impedire la visualizzazione errata della pagina.

```
<NOSCRITP>
sezione per browser che non supportano JS
</NOSCRIPT>
```

Richiamo degli Script

Uno *script* può essere inserito in due modi all'interno di una pagina HTML:

- inserendo il codice nel documento.
- Caricando il codice da un *file* esterno.

Per inserire il codice *JavaScript* direttamente nel documento occorre inserire le istruzioni tra i ***tag*** <SCRIPT> e </SCRIPT> come spiegato nella **pagina precedente**.

```
<HTML>
<HEAD>
<TITLE>Script interno </TITLE>
</HEAD>
<BODY>
<BR> Questa stringa è scritta con l'Html <BR>
<SCRIPT language=JavaScript>
<!-- document.write (Questa con JavaScript); //-->
</SCRIPT>
<BR> Di nuovo HTML.
</BODY>
</HTML>
```
Caricare uno *script* da un *file* esterno può essere utile quando questo deve essere utilizzato su più pagine. Il *file* esterno può essere richiamato tramite un *file* ASCII che avrà estensione .js, la sintassi da inserire all'interno della pagina HTML è la seguente:

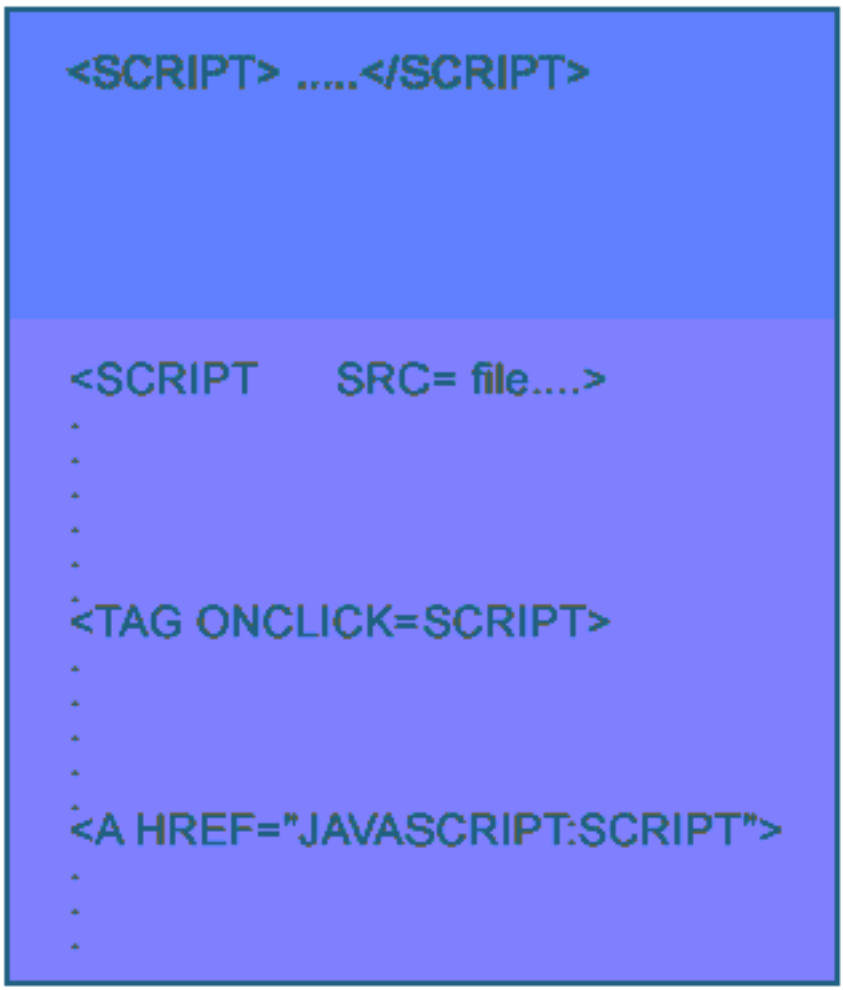
```
<SCRIPT language=JavaScript src=nome_del_file.js>
<!-- //-->
</SCRIPT>
```
Il *file* può essere scritto con qualsiasi *editor* testuale, ma è importante che non contenga ***tag*** di apertura e chiusura degli *script*. In alternativa lo *script* può essere esterno. Scrivere con il blocco note la riga seguente e salvare il documento dandogli il nome prova.js.

```
<HTML>
<HEAD>
<TITLE>Script interno </TITLE>
</HEAD>
<BODY>
<BR> Questa stringa è scritta con l'Html <BR>
<SCRIPT language=JavaScript src=prova.js>
<!-- //-->
</SCRIPT>
<BR> Altro codice HTML.
</BODY>
</HTML>
```

Modalità di esecuzioni

Le istruzioni *JavaScript* possono essere eseguite anche diversamente rispetto ai casi trattati finora, infatti, facendo anche un riferimento anche alle precedenti lezioni, le istruzioni *JavaScript* possono essere eseguite:

- all'interno degli *script* (tra i **tag** <SCRIPT>);
- caricandole da un *file* esterno;
- in seguito all'attivazione di un evento;
- in luogo di un *link*: (da NS 3.0) nella forma <A HREF: *javascript*:comando>;;
- valori *JavaScript* possono essere richiamati dall'HTML includendoli tra i caratteri & { e };%.



Struttura di un documento Javascript

Si può creare uno *script* che preleva l'ora d'inizio del caricamento della pagina sul *client* e la conservi nella **variabile** orainizio. Mettiamo il caso che vogliamo scrivere questo valore in un *textbox*, che può essere visibile all'utente anche dopo molto tempo dall'inizio del caricamento della pagina, per fare ciò ci occorre inserire queste semplici righe:

<INPUT type=text size=10 value=&{orainizio};%></INPUT>
in questo modo il campo visualizzare ha il valore della **variabile** orainizio.

Eventi

Gli eventi sono utilizzati per richiamare le istruzioni. Dato che l'esecuzione degli *script* è sequenziale per inserire della dinamicità all'interno delle pagine occorre che alcune funzioni vengono lanciate solo quando l'utente compie una particolare azione tipo cliccare su un pulsante, completare il *download* di un immagine e così via.

Ad un evento può essere associata un'unica istruzione, ma di solito l'associazione viene fatta con un blocco di istruzioni, le funzioni, che prendono il nome di *handler* o

gestori di eventi. Per interfacciare HTML e *JavaScript* gli eventi non sono inseriti nei ***tag*** <SCRIPT> ma nei ***tag*** dell'HTML. Quando un *browser* compatibile con ***Javascript*** incontra un evento lo interpreta e lo attiva.

Questa è la sintassi generale per creare un *handler* per i ***tag*** HTML:

<TAG onEvento=JavaScript Code>
dove TAG è un ***tag*** dell'HTML compatibile con l'evento, onEvento è il nome dell'evento, e JavaScript Code è la sequenza *JavaScript* che si vuole attivare. Per esempio:

```
<FORM name=prova>
<INPUT type=Text size=15></INPUT>
<INPUT type=Button value=Controlla onClick=Controlla(text.value)></INPUT>
</FORM>
```

Gli eventi si possono attivare anche all'interno degli *script*, come se fossero proprietà dell'oggetto:

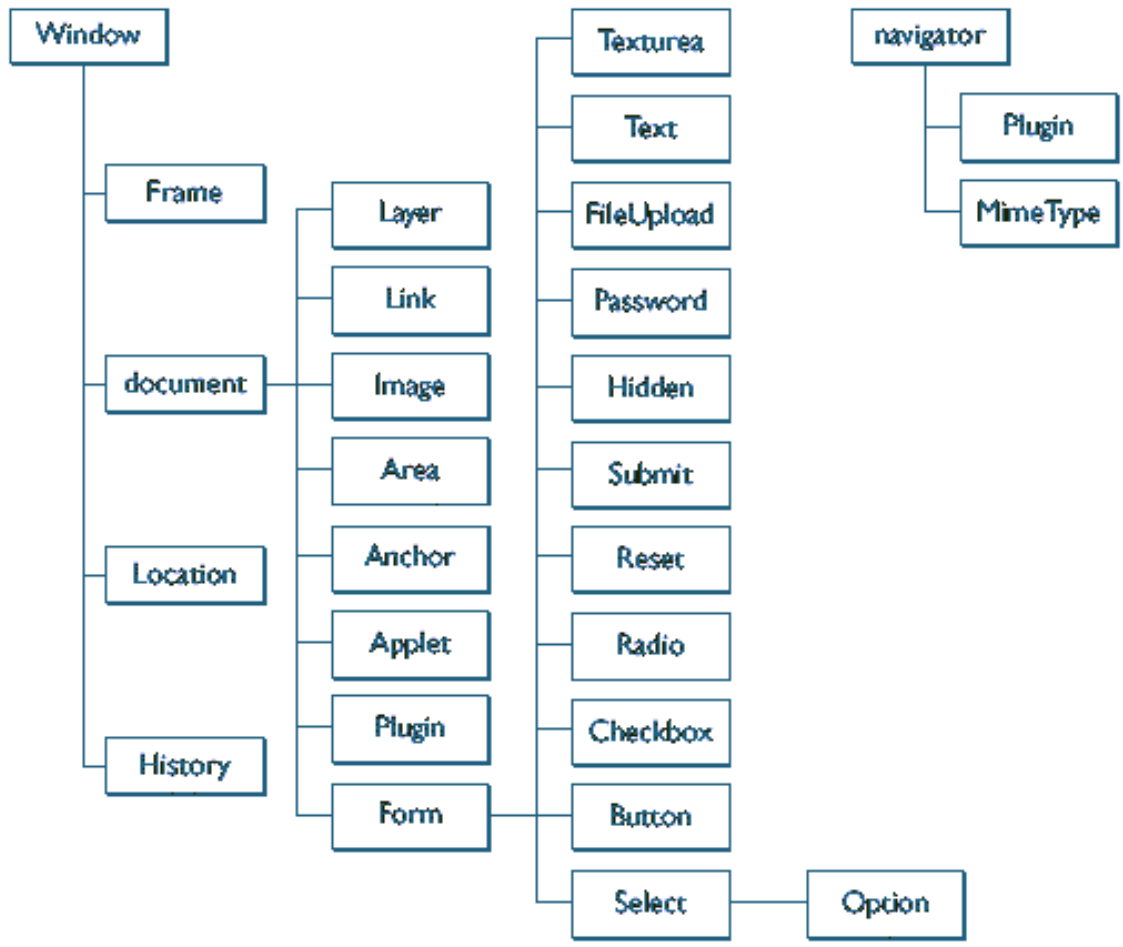
Oggetto.evento=*handler;*
Eventi disponibili

Evento	Si verifica quando	TAG	Versione
onAbort	quando l'utente clicca un *link* o si preme *Stop* nella barra dei comandi del *browser*	IMG	1.1
onBlur	l'oggetto sulla pagina perde il *focus*	SELECT, TEXTAREA, INPUT (TEXT)	1.0
onChange	il contenuto di un campo di un *form* è modificato e non più selezionato	SELECT, TEXTAREA, INPUT (TEXT)	1.0
onClick	*click* su un oggetto o su un *link*.	A, INPUT (tutti)	1.0
onDblClick	doppio *click* del *mouse*	BODY, A	1.2
onDragDrop	*drag & drop* sulla finestra	*Window*	1.2
onError	il caricamento dà un errore	*MG and Window*	1.1
onFocus	un oggetto sulla pagina acquisisce il *focus*	SELECT, TEXTAREA, INPUT (TEXT)	1.0
onKeyDown	viene premuto un tasto	BODY, IMG, A, INPUT (TEXTAREA)	1.2
onKeyPress	si preme e poi rilascia un tasto o lo si tiene premuto	BODY, IMG, A, INPUT (TEXTAREA)	1.2
onKeyUp	tasto precedentemente premuto è stato rilasciato	BODY, IMG, A, INPUT (TEXTAREA)	1.2
onLoad	una pagina o un'immagine finisce il suo caricamento	BODY, FRAMESET	1.0
onMouseDown	si preme un pulsante del *mouse*	BODY, A e i Bottoni	1.2
onMouseMove	si muove il *mouse*	nessuno per *default*	1.2
onMouseOut	il *mouse* esce fuori dall'oggetto	A, Mappe Cliccabili	1.1
onMouseOver	il *mouse* si muove su un oggetto	A, Mappe Cliccabili	1.1
MouseUp	si rilascia un pulsante del *mouse*	A, Mappe Cliccabili	1.1
onMove	si muove una	*Window*	1.2

	finestra o un *frame*		
onReset	il tasto annulla di un *form*	FORM	1.1
onResize	si ridimensiona una finestra	*Window*	1.1
onSelect	selezione di testo	INPUT (TEXT)	1.0
onSubmit	è abbinato al tasto invio del *form*	FORM	1.0
onUnload	si rilascia una finestra	*Window*	1.0

Oggetti Navigatore

Quando viene caricata una pagina nel Navigatore del *Browser* vengono creati un numero di oggetti *JavaScript* settati in base all'HTML e ad altre informazioni pertinenti. La gerarchia di questi oggetti, che rispecchia la struttura di una pagina HTML, è la seguente:



Gerarchia degli oggetti Javascript relativi a una pagina HTML

In questa gerarchia, un oggetto discendente è una proprietà dell'oggetto da cui discende. Per esempio una *form* chiamata *theForm* è un oggetto così come una

proprietà di *document*, ed è referenziata in questo modo: *document.theForm*

Ogni pagina ha i seguenti oggetti:

- ***navigator*:** è utilizzato per acquisire informazioni sul *browser* utilizzato dall'utente, per utilizzare i *plug-in* installati, e per i *MIME* supportati dal *client*;
- ***window*:** le sue proprietà sono destinate completamente alle finestre;
- ***document*:** contiene le proprietà basate sul contenuto del documento come il titolo, *links*, *form*;
- ***location*:** le proprietà basate sull'*URL* corrente;
- ***history*:** contiene la storia di navigazione del *client*.

Per riferirsi ad una specifica proprietà bisogna specificare il nome dell'oggetto e tutti i suoi antenati:

document.theForm.text1.value=prova

Nella precedente stringa si referenzia la proprietà *value* di un campo testo, *text1*, contenuto nel *form theForm* del documento corrente.

Definizione e chiamata di funzioni

Le funzioni in *JavaScript* sono l'elemento portante del linguaggio. Una funzione non è altro che una procedura *JavaScript* capace di compiere una azione specifica. Per definire una funzione occorrono quattro componenti:

- la *Keyword function*.
- Il nome della funzione.
- Gli argomenti della funzione compresi tra le parentesi tonde e separati dalle virgole.
- Le istruzioni *JavaScript* comprese tra le parentesi graffe.

Grazie alle funzioni è possibile scrivere codice più conciso, infatti, si può scrivere un gruppo di istruzioni, assegnarvi un nome e quindi eseguire l'intero gruppo in qualsiasi momento richiamandolo e specificando le informazioni necessarie. Le informazioni da passare alla funzione devono essere specificate tra parentesi tonde dopo il nome della funzione. Di solito le funzioni vengono inserite all'interno del documento nella sezione *HEAD* in modo che questa possa essere caricata subito e resa sempre disponibile all'interno della pagina.

```
<HEAD>
<SCRIPT LANGUAGE=JavaScript>
<!--
function quadrato(x) {
return x*x;
}
//-->
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT>
```

document.write (La funzione ritorna , quadrato(5), .);
</SCRIPT>
<P>OK?</P>
</BODY>
La funzione definita prende il nome di quadrato, possiede un unico parametro x, è composta da un unica istruzione: *return* x*x. La funzione poi viene richiamata all'interno della sezione BODY, semplicemente specificando il nome della funzione e il valore del parametro (il parametro passato può essere anche una **variabile**).

JavaScript possiede una serie di funzioni predefinite che sono le seguenti:

- *Escape*.
- *Eval*.
- *isFinite*.
- *isNaN*.
- *Number*.
- *parseFloat*.
- *parseInt*.
- *String*.
- *Taint*.
- *Unescape*.
- *UnTaint*.

Variabili, valori e letterali

JavaScript riconosce i seguenti tipi di valore:

- **Numeri** (interi o decimali).
- **Valori Logici** (*Boolean*). Può avere due stati: *true* e *false*. Nei confronti le espressioni con risultati 0 vengono considerate false, mentre le istruzioni che danno come risultato un numero diverso da 0 sono considerate vere.
- **Stringhe**, come Pippo. Una stringa contiene zero o più caratteri racchiusi tra virgolette semplici (' ') o doppie(''). La stringa deve essere delimitata dallo stesso tipo di virgoletta.
- **Null**.

JavaScript è *case-sensitive*, per cui *null* è diverso da *Null*, come *myvar* è diversa da *myVar*. Quando si dichiarano le variabili non occorre specificare il tipo di dato che andrà a contenere: verrà fatto a seconda dell'assegnamento:

var myVar=24
Assegna a *myVar* il valore 24 e questa **variabile** viene definita automaticamente di tipo numerica. Se nel corso dello *script*, contenente l'istruzione si aggiunge la riga:

myVar=Questa è una stringa
Non viene generato nessun errore: la **variabile** *myVar* generata per prima (*var myVar*=24) sarà considerata di tipo numerico, mentre la seconda sarà considerata di tipo *String*, e la prima istanza viene persa. Non tutti i nomi possono essere assegnati ad una **variabile**, infatti, si devono rispettare alcune direttive:

- il nome della **variabile** deve iniziare con una lettera o con il carattere _

(trattino basso).
- può contenere le cifre numeriche (0-9).
- può includere le lettere comprese da A fino Z e le lettere comprese da a fino z estremi esclusi, c'è differenza tra la prima serie e la seconda in quanto *JavaScript* è *case-sensitive*.

Operatori

Gli operatori possono essere unari o binari, i primi richiedono solamente un operando al contrario dei binari che ne vogliono due. ***Javascript*** ha questi tipi di operatori:

- Assegnamento.
- Confronto.
- Aritmetici.
- *Bitwise*.
- Logici.
- Stringhe.

Operatori di Assegnamento

Assegna il valore dell'operando a destra dell'operatore all'operando presente alla sinistra dell'operatore. L'operatore di assegnamento base è l'uguale (=). La tabella seguente contiene gli altri operatori di assegnamento e sono riportati sia con la forma abbreviata che con quella integra.

Forma abbrevviata	Forma integra
x+=y	x=x+y
x-=y	x=x-y
x*=y	x=x*y
x%=y	x=x%y
x<<=y	x=x<<y
x>>=y	x=x>>y
x>>>=y	x=x>>>y
x&=y	x=x&y
x^=y	x=x^y
x|=y	x=x|y

Operatori di confronto

Compara due operandi e ritorna un valore logico a seconda dell'esito del confronto. Se l'esito è positivo ritorna 1, altrimenti 0. Gli operatori di confronto sono i seguenti:

- == vero se gli operandi sono uguali.
- != vero se gli operando non sono uguali.
- > vero se l'operando di sinistra è maggiore di quello destro.
- >= vero se l'operando di sinistra è maggiore o uguale di quello destro.
- < vero se l'operando di sinistra è minore di quello destro.
- <= vero se l'operando di sinistra è minore o uguale di quello destro.

Operatori di Aritmetrici

Prelevano dei valori numerici per elaborarli e ritornare un singolo valore numerico.

- + (somma).
- - (sottrazione).
- * (moltiplicazione).
- / (divisione).
- % (resto intero).

Altri operandi aritmetici, sono l'incremento e il decremento e il meno unario. Questi al contrario dei precedenti sono operatori unari.

- ++ Incrementa di un unità.
- -- Decrementa di un unità.
- - Rende negativo un numero.

Operatori Logici

Ritornano due valori: 0 se l'espressione logica è vera, 1 se l'espressione è falsa. Gli operatori logici sono:

- ***expr1 && expr2*** *(and)* vero solo se entrambi gli operandi sono veri.
- ***expr1* || *expr2*** *(or)* vero se almeno uno degli operandi è vero.
- ***!expr*** *(not)* è la negazione dell'argomento.

Autenticazione ed integrità dei dati

Cosimo Laneve

16.3.1 (Implementare appropriate misure di sicurezza in un sito Web.)

Integrità dei messaggi

Integrita' dei messaggi

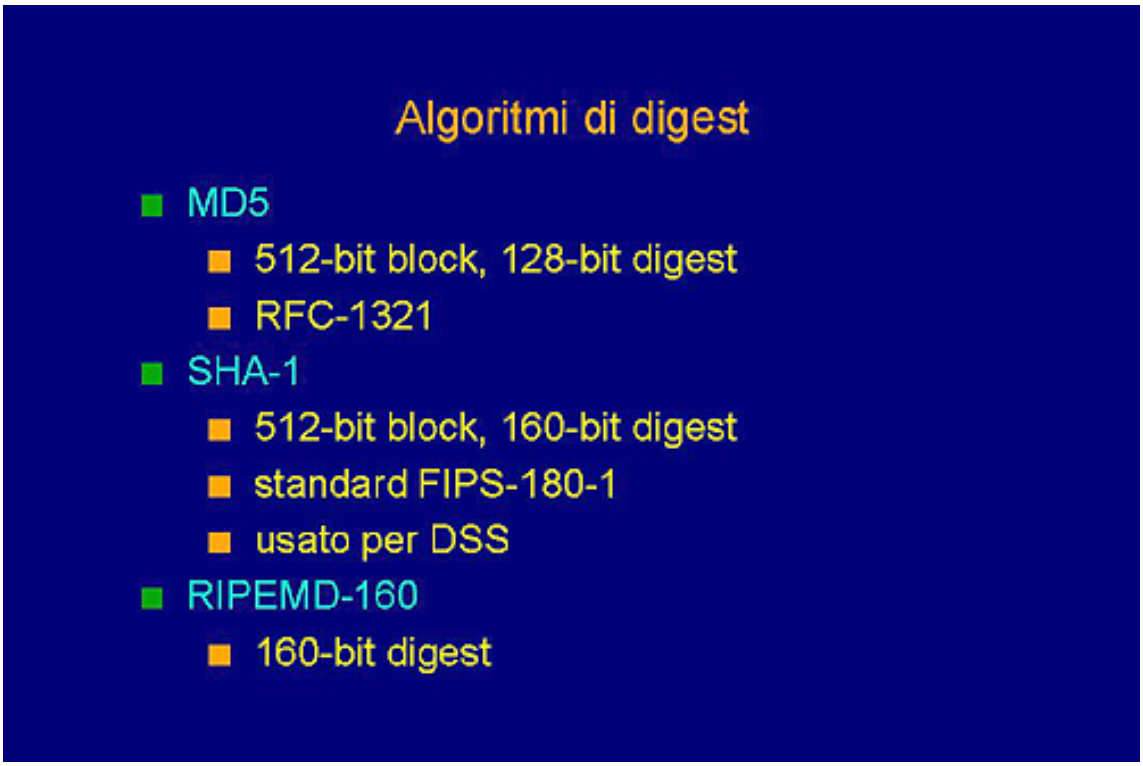
Una delle proprietà di sicurezza che desideriamo avere all'interno di un sistema informatico, è quella che riguarda l'integrità dei messaggi che vengono scambiati e dei dati. Questo perché, anche nel caso che noi cifriamo una comunicazione e quindi le persone che eventualmente le intercettassero non possono leggerlo, possono ciononostante cambiare dei bit della comunicazione cifrata e questo può dar luogo a dei messaggi decifrati di tipo completamente imprevedibile. È chiaro che se chi riceve il messaggio che è stato decifrato in maniera scorretta è un essere umano, probabilmente se ne accorgerà e chiederà che il messaggio venga ritrasmesso. Ma nel caso, invece, che il messaggio sia destinato ad un sistema di elaborazione, che automaticamente deve svolgere delle procedure o delle operazioni, se i dati su cui lavora sono sbagliati, è molto probabile che anche il lavoro che lui cercherà di svolgere sarà di tipo sbagliato e potrebbe addirittura causare dei danni seri a dei sistemi fisici.

Message digest (hash)

Message digest (hash)

Per evitare che i dati vengano danneggiati, sia durante la trasmissione sia mentre sono parcheggiati su disco, si tende ad utilizzare dei codici di protezione basati sui cosiddetti algoritmi di *message digest* o algoritmi di *hash*. L'algoritmo di *message digest* è a tutti gli effetti un riassunto del contenuto del messaggio che si vuole proteggere. È quindi possibile confrontare il riassunto costruito a partire dal messaggio originale con il riassunto ottenuto dal messaggio che è stato trasmesso e vedere se questi due sono identici. Un buon algoritmo di *message digest* deve essere veloce da calcolare per non sovraccaricare troppo i sistemi su cui viene utilizzato e, soprattutto, deve essere difficile da invertire, ossia deve essere pressoché impossibile, partendo dal *digest*, ricostruire il messaggio originale. Molto spesso gli algoritmi di *digest* vengono utilizzati non soltanto per proteggere i dati, ma anche in unione con algoritmi a chiave pubblica, perché, visto che questi algoritmi sono lenti, facendoli operare non sui dati originali ma sul loro riassunto, questi algoritmi diventano più veloci.

Algoritmi di digest



Algoritmi di digest

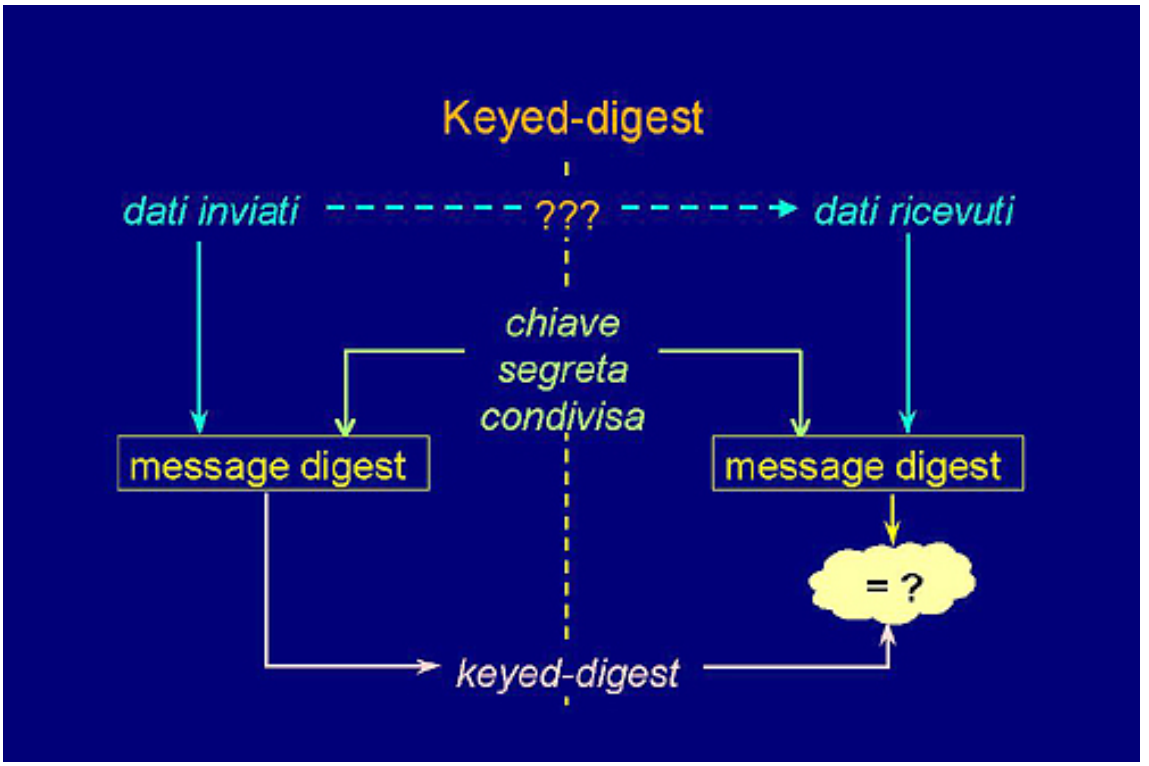
Esistono moltissimi algoritmi di *digest*, quelli più usati sono attualmente tre: l'algoritmo MD5, opera su blocchi dati da 512 bit e genera un riassunto da 128 bit. L'algoritmo SHA-1, che è quello utilizzato dal governo americano all'interno del suo sistema di firma digitale, opera anch'esso su blocchi da 512 bit, ma genera un *digest* di dimensione maggiore, da 160 bit. Talvolta viene anche utilizzato il *digest* RIPEMD-160, che genera *digest* da 160 bit ed è stato sviluppato in Europa all'interno del progetto RIPE. In generale è opportuno utilizzare algoritmi che generino dei riassunti lunghi, perché si può dimostrare matematicamente che più il riassunto generato è lungo, ossia composto da un maggior numero di bit, e meno sono le informazioni che si perdono, maggiore è la resistenza agli attacchi crittografici che possono essere condotti.

MAC, MIC, MID

MAC, MIC, MID

In generale questo riassunto, in una forma opportuna, viene aggiunto al messaggio. Normalmente questa aggiunta prende il nome di MIC, oppure di MAC. Si chiama MIC nel caso in cui chi ha aggiunto questo codice al messaggio, che viene trasmesso o memorizzato, intendesse sottolineare maggiormente il fatto che questo codice permetta di garantire l'integrità del messaggio. Ma poiché spesso l'integrità non è utile senza avere simultaneamente l'autenticazione di chi ha generato i dati, il codice talvolta viene chiamato MAC (*Message Authentication Code*), volendo così sottolineare che fornisce non solo integrità ma anche autenticazione dei dati. Visto che stiamo aggiungendo un codice ai nostri dati, generalmente quello che capita è di utilizzare questo codice per introdurre anche dei dati aggiuntivi, tipicamente un *message identifier*, ossia un numero di serie che identifichi questo come il messaggio, ad esempio, numero 27 e poi 28, 29 e 30. In questo modo siamo in grado di parare attacchi di tipo *replay*, in cui un medesimo messaggio venga inviato più volte, o attacchi di tipo cancellazioni, in cui un messaggio venga cancellato dal normale flusso dei dati.

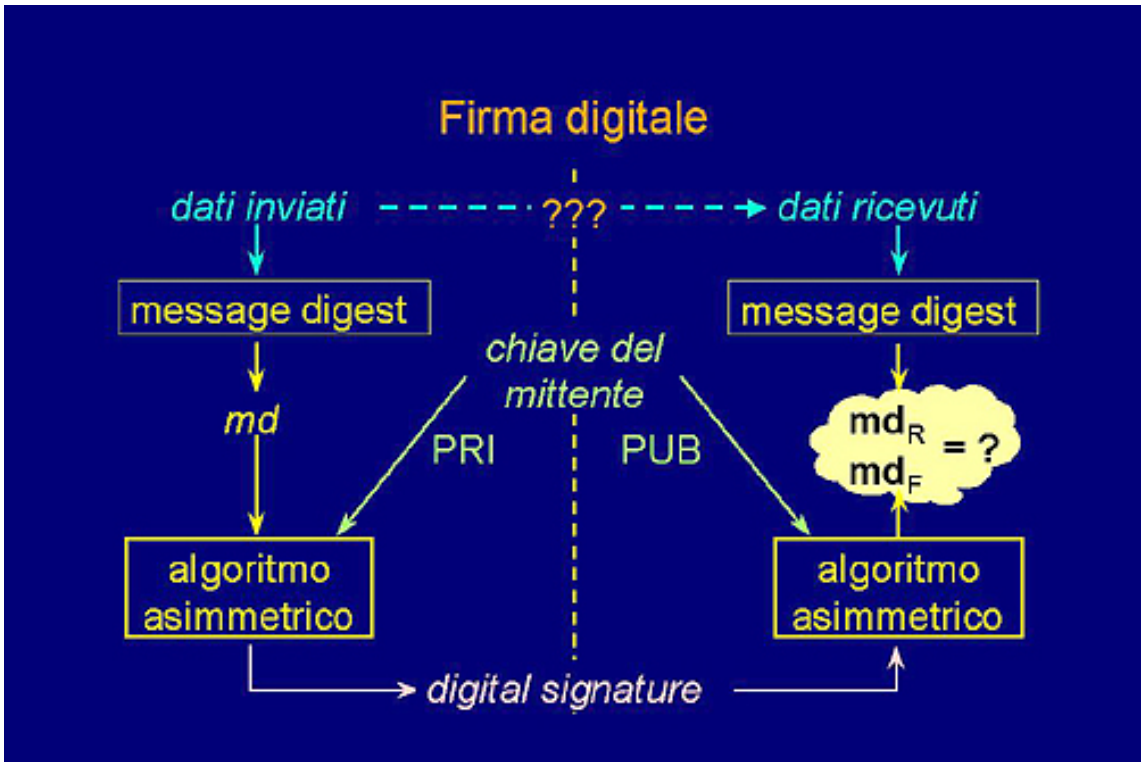
Keyed digest



Keyed digest

Ovviamente non ha nessun senso spedire direttamente un messaggio e il suo riassunto, perché un eventuale attaccante provvederebbe sia a cambiare il messaggio sia a ricalcolare il riassunto, in modo da non farsi accorgere dell'azione che ha appena svolto. È quindi chiaro che si manda il messaggio, ma bisogna mandare il *digest* sotto forma protetta, in particolare esistono due principali tecniche per proteggere il *digest*. Quando le prestazioni sono di grande importanza, ad esempio nella trasmissione dei dati su reti molto veloci, occorre avere delle funzioni di *digest* altrettanto veloci e delle funzioni di protezione che hanno la medesima proprietà. In questo caso si utilizza il cosiddetto *keyed-digest*: vengono inviati i dati e una volta ricevuti c'è un ragionevole dubbio se qualcuno ha manipolato i dati durante la loro trasmissione. Per effettuare questo controllo si procede come segue: i dati inviati sono anche passati dentro un algoritmo di *digest* assieme a un segreto, che è condiviso tra mittente e destinatario. Il risultato dell'algoritmo dipende sia dai dati sia dalla chiave e costituisce il cosiddetto *digest* con chiave (*digest* protetto con chiave o *keyed-digest*). Chi riceve i dati svolgerà la medesima operazione sui dati ricevuti, ossia li introdurrà nell'algoritmo di *digest* assieme alla chiave segreta nota anche a lui e otterrà il riassunto calcolato sui dati ricevuti. Dopodiché, confronterà se questo riassunto calcolato sui dati ricevuti è uguale al *keyed-digest* che ha ricevuto, se le due cose sono uguali possiamo essere certi che i dati sono corretti, nessuno li ha manipolati ed arrivano dalla persona che condivide con noi questa chiave segreta. Nel caso, invece, che i due *digest* non coincidano, purtroppo non possiamo capirne la causa, ossia non riusciamo a distinguere il caso in cui i dati siano stati manipolati dal caso in cui sia stato manipolato, o falsificato, il *digest*. Non è possibile distinguere queste due cause di fallimento.

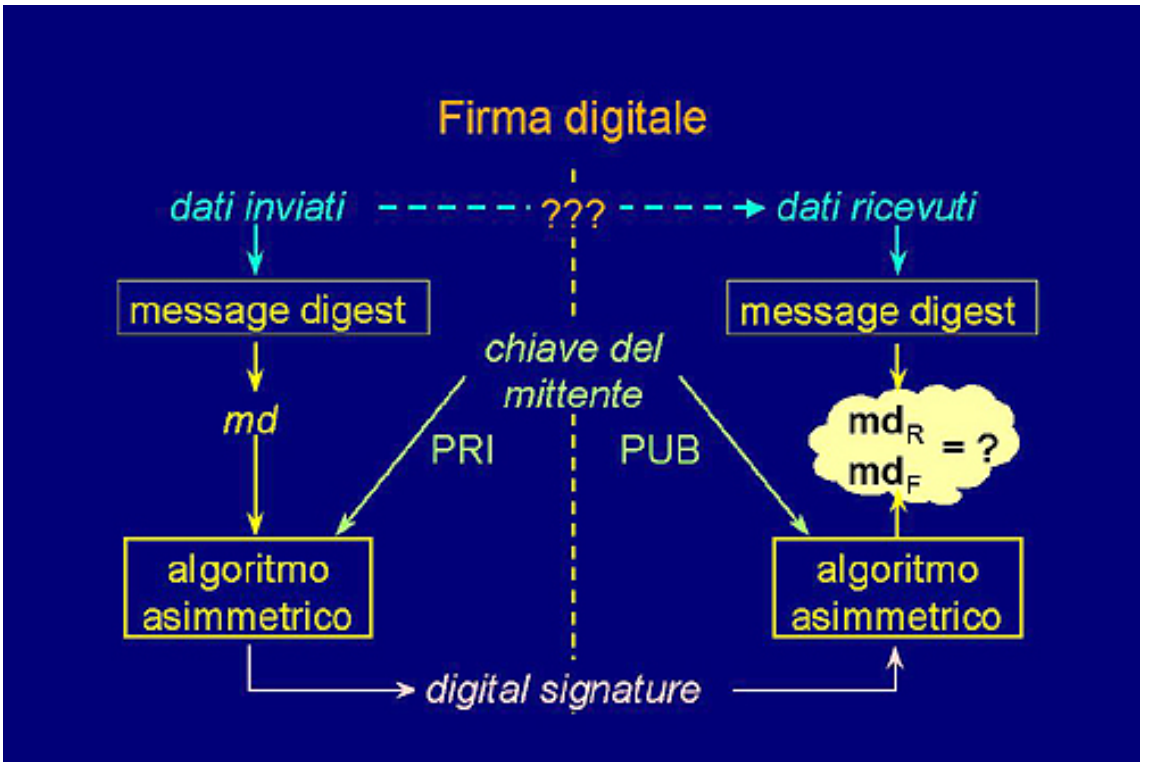
Firma digitale 1



Firma digitale 1

Se l'integrità dei dati ci serve non soltanto per nostro uso interno, ma ci serve per poter dimostrare formalmente in una causa legale, in tribunale, o comunque in modo inoppugnabile, che certi dati sono stati prodotti da una certa persona, allora il *digest* viene normalmente protetto non mediante una chiave condivisa, ma mediante una chiave tipica di ciascun individuo, ossia con la sua chiave privata e quindi utilizzando algoritmi di tipo asimmetrico. Questo costituisce a tutti gli effetti la vera e propria firma digitale secondo gli standard legali esistenti in Italia e in molti altri paese del mondo. La firma digitale opera come segue: prendiamo i dati e li inviamo a qualcuno, questa persona ha il ragionevole dubbio che i dati siano stati manipolati durante la trasmissione. Per rassicurarla quello che possiamo fare è far passare i dati attraverso un algoritmo di *digest* e così calcolare il suo riassunto. Dopodiché, prendere questo riassunto e cifrarlo mediante la nostra chiave privata. Questo costituisce tecnicamente la firma digitale del messaggio, ossia la firma digitale del messaggio è il suo riassunto cifrato con la chiave privata del mittente.

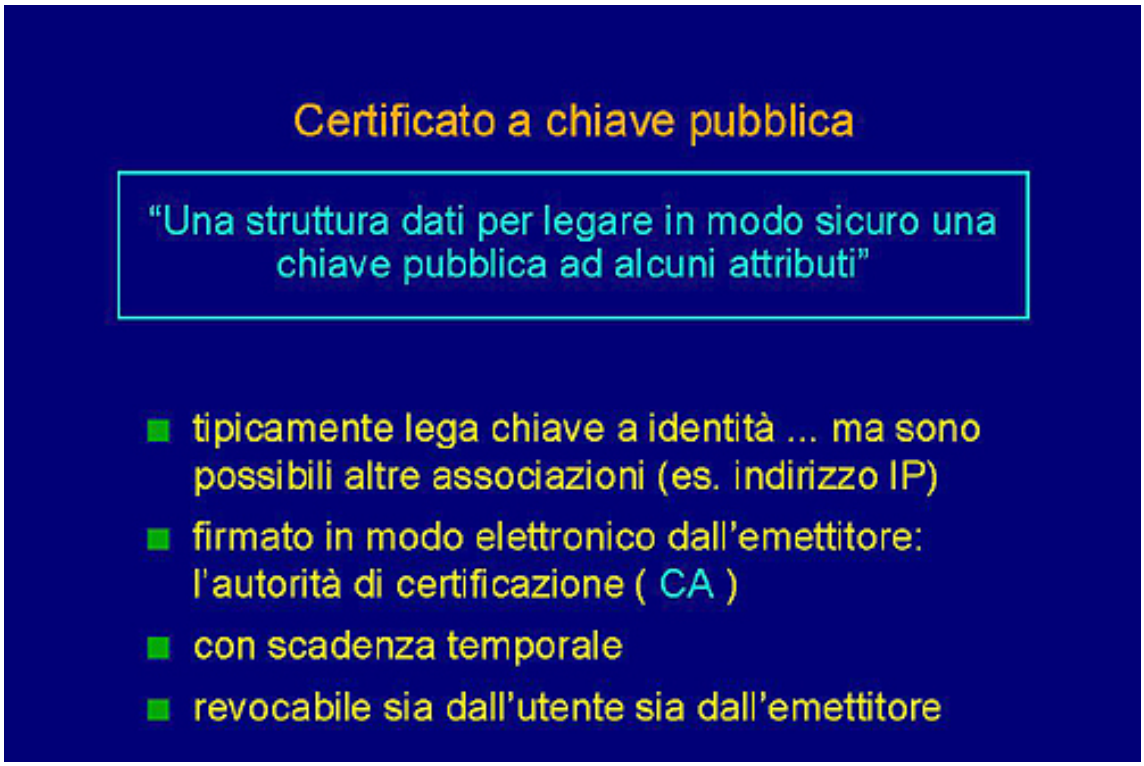
Firma digitale 2



Firma digitale 2

Chi riceve i dati e riceve anche la firma digitale, può verificare se i dati sono corretti attuando il seguente procedimento: prende i dati e ne calcola il *digest*, ottenendo il riassunto calcolato sui dati ricevuti. Dopodiché, a partire dalla firma digitale ottiene il *digest* estratto dalla firma, perché in grado di decifrarla utilizzando la chiave pubblica del mittente. Se questi due *digest* coincidono, allora abbiamo la certezza di due cose: che i dati ricevuti sono integri e che questi dati sono stati generati dalla persona che possiede la chiave privata corrispondente alla chiave pubblica da noi utilizzata. Se invece i due *digest* non coincidono, non abbiamo modo di capire la causa: può essere stato causato da manipolazioni durante la trasmissione dei dati, da una manipolazione durante la trasmissione della firma digitale, oppure semplicemente dal fatto che abbiamo usato la chiave pubblica sbagliata, ossia i dati non arrivano da chi noi crediamo essere l'originatore del messaggio. Sottolineiamo una volta di più che la firma digitale dipende sia dai dati che dal mittente. In particolare, se per caso i dati vengono cambiati anche soltanto in una parte non significativa, come ad esempio un aumento del numero di spazi tra due caratteri, la firma digitale immediatamente ci avverte di questa manipolazione.

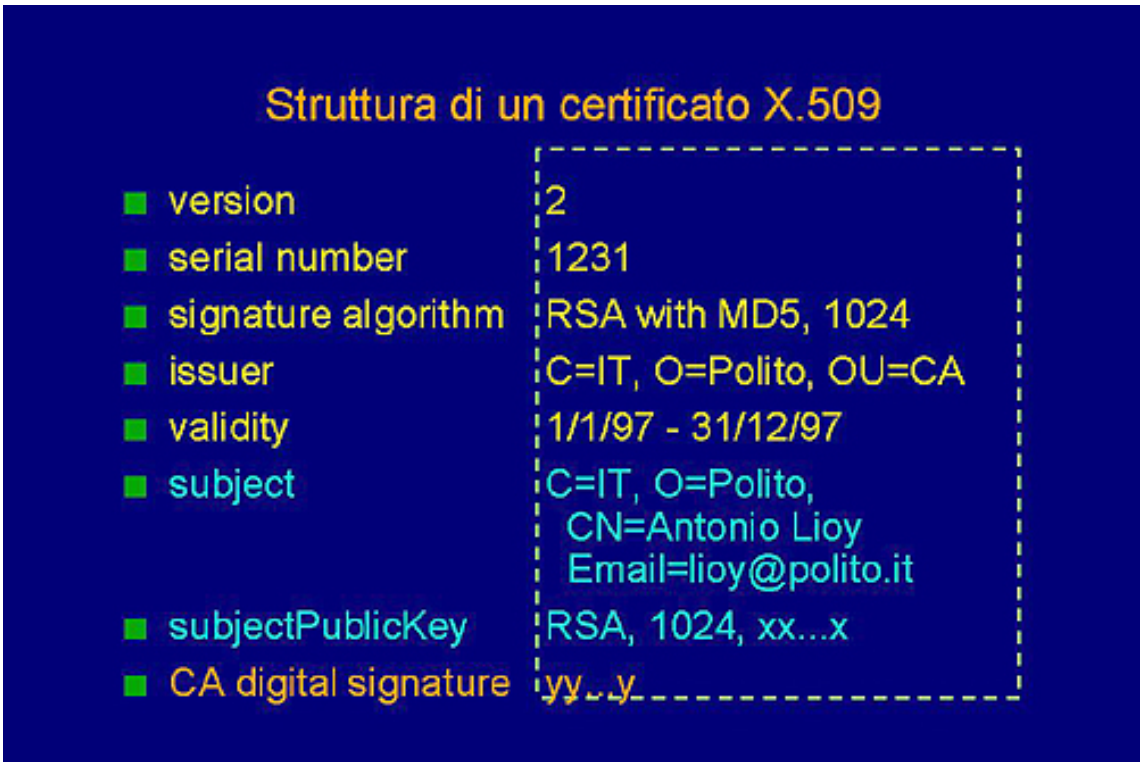
Certificato a chiave pubblica



Certificato a chiave pubblica

Per sapere a chi appartiene una certa chiave pubblica, per poter ad esempio attribuire con certezza una firma digitale ad una persona, bisogna disporre di un certificato a chiave pubblica. Il certificato a chiave pubblica è definito come una struttura dati che lega in modo sicuro una chiave pubblica con alcuni attributi. Siccome tipicamente questi documenti con firme digitali vengono usati da esseri umani, è abbastanza tipico che il certificato leghi la chiave con l'identità della persona. Ma poiché le chiavi pubbliche e private possono essere utilizzate da qualunque elemento attivo di un sistema di elaborazione, sono possibili anche altre associazioni. Ad esempio si può dire che una certa chiave pubblica corrisponde alla chiave privata utilizzata da un nodo di elaborazione che ha un certo indirizzo IP. Per evitare che il certificato a chiave pubblica venga manipolato dopo che è stato emesso, viene firmato tramite firma digitale dall'emettitore. L'emettitore si chiama autorità di certificazione ed è l'autorità che si fa garante dell'associazione fra chiave pubblica e attributi. Un certificato digitale ha delle proprietà simili a quelle di molti altri certificati, tipo carta d'identità o passaporto che dir si voglia. Ad esempio, ha una scadenza temporale. Inoltre è revocabile sia da chi ha emesso il certificato sia dall'utente, nel caso che qualcosa sia cambiato, ad esempio nel caso di furto della chiave privata.

Struttura di un certificato X.509



Struttura di un certificato X.509

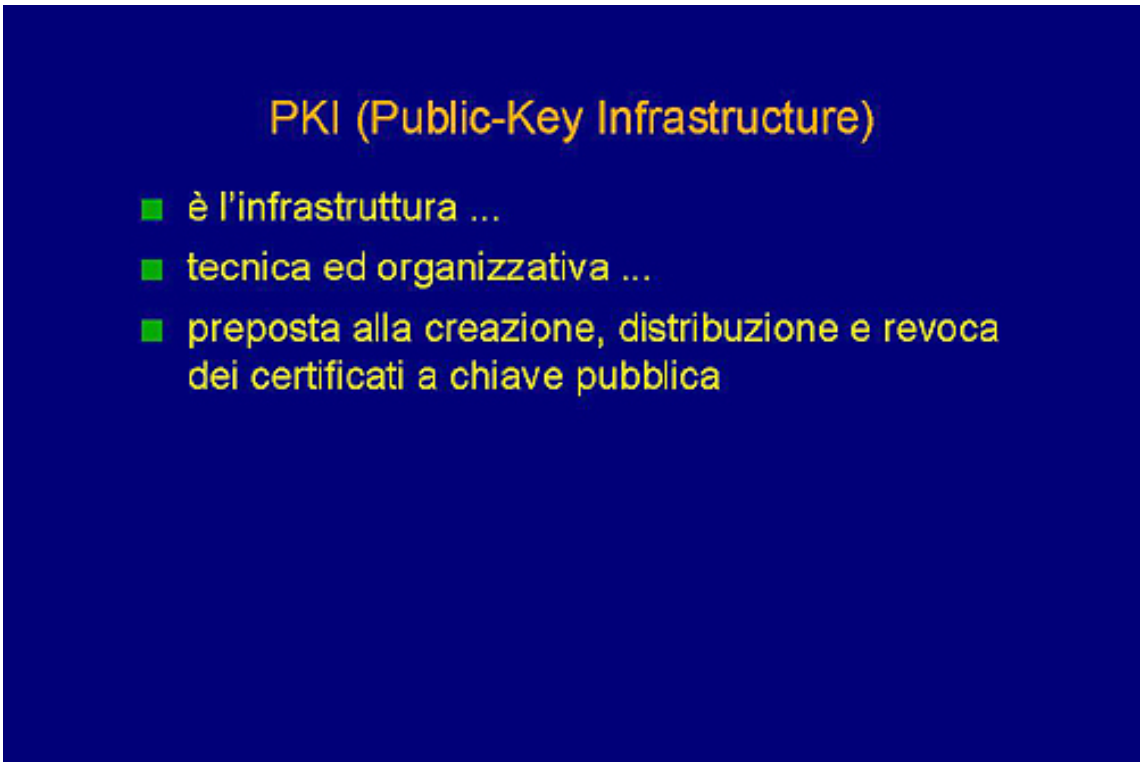
I certificati a chiave pubblica oggigiorno più utilizzati sono i certificati che seguono lo standard X.509. In particolare lo standard X.509 versione 3. Qui viene riportato un esempio di alcuni dati contenuti in un certificato X.509 versione 3. All'inizio ci sono alcuni dati tecnici: tipo la versione (nonostante qui ci sia scritto 2 questo è un certificato versione 3, perché la numerazione parte da 0), alcuni dati tecnici sull'algoritmo con cui è stato firmato questo documento (algoritmo RSA applicato ad un *digest* MD5) e la chiave RSA era da 1024 bit. Questo certificato ha numero di serie 1231 per distinguerlo da altri certificati emessi dallo stesso emettitore. Gli emettitori molto spesso sono identificati tramite una notazione che inizia con C, ad indicare *country* = nazione (Italia), prosegue con O per dire organizzazione (Politecnico di Torino) e termina con OU, unità organizzativa, in particolare qui si sta parlando dell'autorità di certificazione del politecnico di Torino. Ogni certificato ha una validità, che è espressa tramite la data di inizio e la data di fine, durante la quale la CA si fa garante dell'associazione. L'associazione è specificata subito dopo. In questo caso la CA si fa garante che questa chiave pubblica RSA da 1024 bit, i cui bit sono elencati qui di seguito, corrisponde alla persona chiamata Antonio Lioy (CN indica nome comune del politecnico di Torino in Italia) e si fa anche garante che questa persona corrisponde all'indirizzo di posta elettronica lioy@polito.it. Per evitare che dopo l'emissione qualcuno cambi uno qualunque di questi dati, la CA appone la sua firma digitale e questo rende il certificato inalterabile, o meglio, se il certificato venisse alterato la manipolazione verrebbe immediatamente rivelata.

X.509 CRL (Certificate Revocation List)

X.509 CRL (Certificate Revocation List)

Può capitare che a una persona venga rubata, o la persona smarrisca, la propria chiave privata. In questo caso non è più vero quanto attestato dal certificato, non è più vero che il certificato contiene la chiave pubblica corrispondente alla chiave privata di una certa persona. In questo caso il certificato deve essere dichiarato non più valido. Il metodo con cui i certificati vengono dichiarati non più validi è quello di includerli in una lista chiamata CRL. La CRL è la lista dei certificati revocati, ossia dei certificati non più validi. Le CRL sono numerate, anzi gli viene attribuito una data, questa è la CRL del 30 settembre del 1999 e dice che il certificato numero 104 non è più valido a partire dal 25 aprile 1998, perché si è rotta la *smart-card* entro cui la chiave privata era contenuta. Invece il certificato 97 non è più valido a partire dal 30 agosto 1999, a causa di un furto del portatile che conteneva la chiave privata. Per evitare che qualcuno emetta delle CRL false, anche la CRL viene firmata dalla autorità di certificazione che ha emesso la CRL. Si noti che l'autorità di certificazione agisce in modo abbastanza piratesco, nel senso che l'autorità di certificazione emette questo elenco di certificati non più validi, ma demanda agli utilizzatori dei certificati il controllo. Ogniqualvolta si riceve una firma digitale accompagnata da un certificato a chiave pubblica, è compito di chi riceve questa firma andare a verificare che il certificato associato sia ancora valido, ossia non sia inserito all'interno di una lista di certificati revocati.

PKI (Public-Key Infrastructure)



PKI (Public-Key Infrastructure)

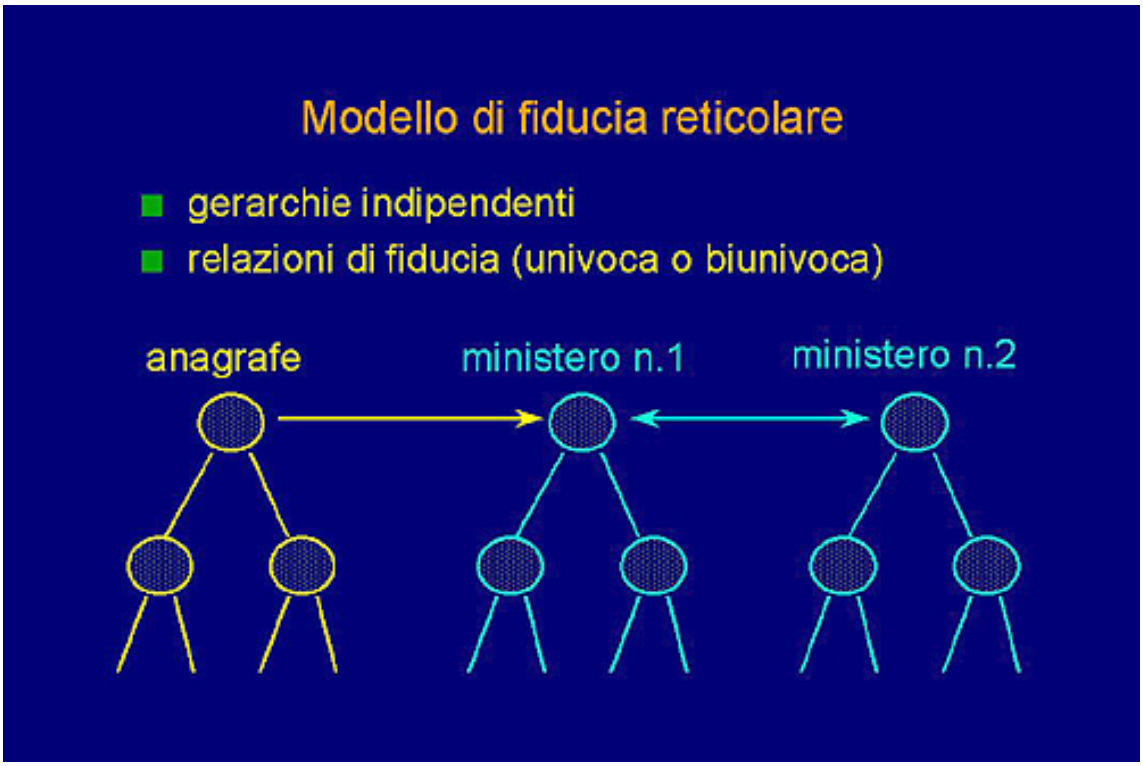
Ovviamente bisogna creare una infrastruttura, chiamata PKI (infrastruttura a chiave pubblica), che è una infrastruttura tecnica ed organizzativa per creare, distribuire e revocare questi certificati. Questo ci permetterà di sfruttare a pieno tutti i benefici degli algoritmi asimmetrici, ossia degli algoritmi a chiave pubblica. In particolare una PKI è costituita da una o più autorità di certificazione.

Certification Authority (CA)

Esempio: SSI per Apache (2)

L'autorità di certificazione è formalmente, diciamo tecnicamente, un'autorità in cui un insieme di utenti ripone la propria fiducia per creare e assegnare certificati a chiave pubblica. Nello standard X.509 non viene specificata nessuna relazione fra l'autorità di certificazione, l'emettitore e il soggetto specificato nel certificato. Esistono però delle norme di legge che dicono che se una firma digitale deve avere valore legale nei confronti di chiunque, questa deve essere emessa non da una generica autorità di certificazione, ad esempio interna ad un'azienda, ma ad un'autorità di certificazione che risponde a certi requisiti legali. È ovviamente possibile creare sia delle autorità di certificazione indipendenti, che permetteranno una comunicazione sicura solo fra tutti gli utenti che si fidano di quella particolare autorità di certificazione, oppure creare delle reti di autorità di certificazione, ossia delle autorità di certificazione che si riconoscono reciprocamente. Il modo più semplice di organizzare queste reti è in strutture gerarchiche: ossia permettere a un'entità suprema, una CA di primo livello, di certificare quindi di asserire la sua fiducia in autorità di certificazione di secondo livello e così a scendere fino alle autorità di certificazione di più basso livello, che emetteranno certificati per i nostri utenti.

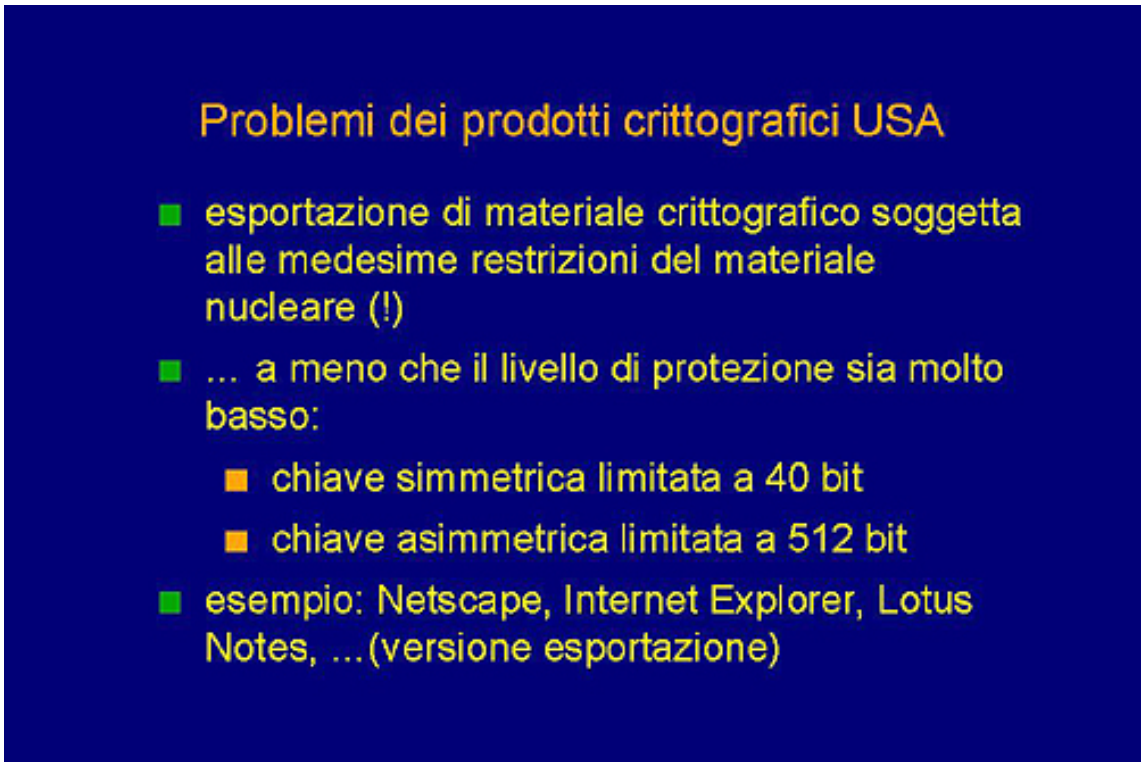
Modello di fiducia reticolare



Modello di fiducia reticolare

Una autorità di certificazione gerarchica sarebbe l'ideale dal punto di vista tecnico. Purtroppo, a causa degli innumerevoli litigi che esistono fra gli esseri umani, non sempre è possibile accordarsi su un'unica struttura di certificazione. Oggigiorno la situazione reale delle infrastrutture di certificazione corrisponde piuttosto allo schema che si vede qui illustrato. Esistono delle autorità di certificazione gerarchiche limitatamente a certe funzionalità o certe aziende. Dopodiché esisteranno tante gerarchie: nel caso che un messaggio dotato, ad esempio, di firma digitale viaggi da una gerarchia all'altra, non verrebbe riconosciuto automaticamente. Per farlo riconoscere bisogna che i gestori di queste infrastrutture di certificazione stabiliscano delle relazioni di fiducia. Le relazioni di fiducia possono essere univoche o biunivoche. Per esempio, l'anagrafe potrebbe fidarsi univocamente dei certificati emessi dal ministero n.1, ma questo ministero potrebbe non aver fiducia nei certificati emessi dall'anagrafe. Viceversa, fra due ministeri appartenenti al medesimo governo si potrebbe avere una relazione di fiducia reciproca e quindi, teoricamente, potrebbero essere collocati sotto una unica autorità di certificazione centralizzata.

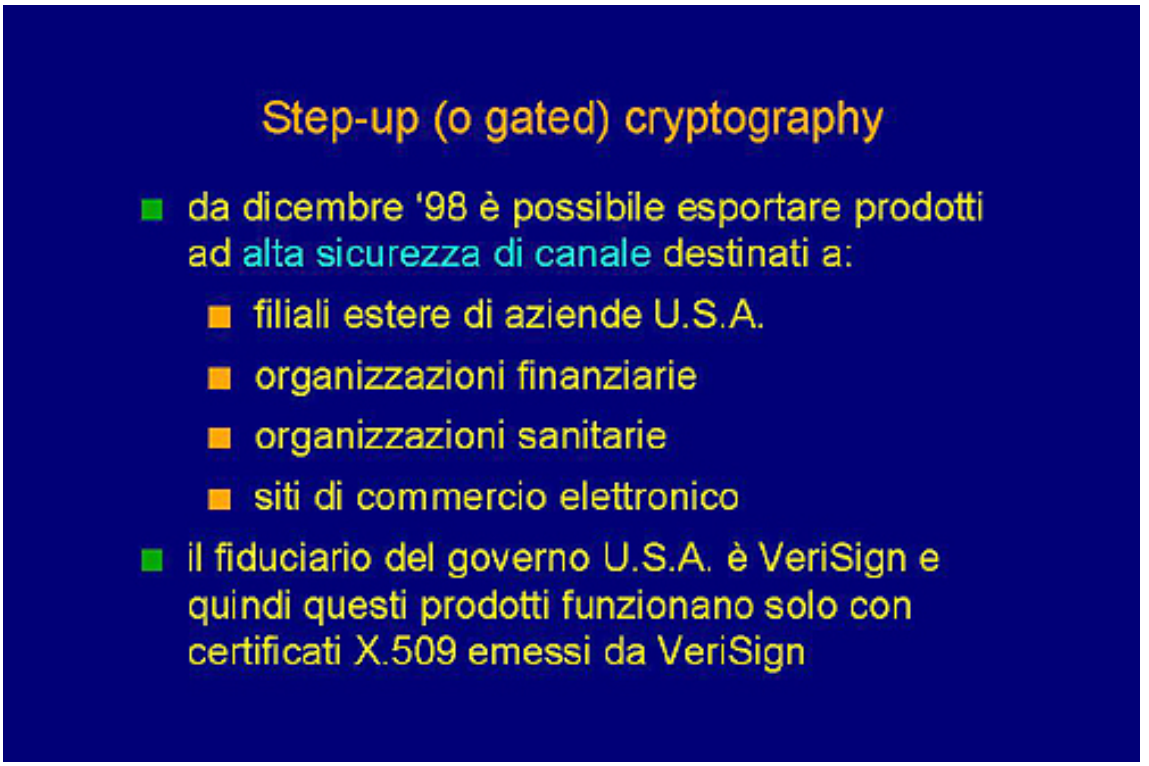
Problemi dei prodotti crittografici USA



Problemi dei prodotti crittografici USA

Terminiamo questa chiacchierata segnalando un particolare problema, che più che essere un problema di tipo tecnico è un problema di tipo politico e organizzativo. Gli stati Uniti, come ben si sa, sono i *leader* mondiali nel campo della tecnologia informatica e anche nel campo della sicurezza offrono sicuramente alcuni fra i migliori prodotti al mondo. Purtroppo il governo americano ritiene che l'esportazione di prodotti crittografici possa danneggiare gravemente la propria sicurezza e quindi impedisce alle proprie aziende di esportare prodotti che contengano sistemi di sicurezza forti. Le aziende sono libere di esportare questi prodotti solo ed esclusivamente se abbassano il livello di protezione. In particolare si impone un limite di 40 bit sulla chiave simmetrica utilizzata da questi prodotti e un limite di 512 bit sulle chiavi asimmetriche. Questo significa che anche quei prodotti di largo uso in tutti i paesi del mondo, tipo i *browser Netscape* o *Explorer* e i sistemi di *workflow* quali *Lotus Notes*, che contengono al loro interno dei sistemi di sicurezza, sono tutti quanti in versione esportazione. Non è la versione identica a quella che si trova negli Stati Uniti, ma sono delle versioni molto deboli perché, per quanto abbiamo detto nelle lezioni precedenti, delle chiavi simmetriche inferiori a 64 bit, o delle chiavi asimmetriche inferiori a 1024 bit, sono da reputarsi altamente insicure.

Step-up (o gated) cryptography



Step-up (o gated) cryptography

Nel dicembre 1998 il governo americano ha leggermente cambiato la propria opinione in materia e ha permesso l'esportazione di prodotti di sicurezza forti, ossia pari a quelli disponibili in America, nel caso che questi prodotti non realizzino funzionalità di cifratura dati in dispositivi di memorizzazione, ma solo funzionalità di sicurezza durante il transito di dati in rete. Quindi ci permettono di proteggerci dagli attacchi condotti in rete, non dagli attacchi o dalle indagini condotti direttamente sui nostri calcolatori. Il vincolo è caduto e quindi ora è possibile esportare prodotti ad alta sicurezza di canale per proteggere la *privacy* dei dati, o i siti di commercio elettronico. Tutto questo però, solo con destinatari specifici, come filiali estere di aziende statunitensi, organizzazioni finanziarie, banche, assicurazioni e organizzazioni sanitarie. Esiste però un ulteriore vincolo: questi prodotti vengono venduti e diventano operativi solo se utilizzano certificati X.509 emessi dal fiduciario del governo statunitense. Il fiduciario del governo statunitense è l'autorità di certificazione chiamata *VeriSign*. Quindi, se qualcuno appartiene a una di queste categorie qui elencate e desidera comprare un prodotto di questo genere, non può dotare quel prodotto di un certificato X.509 emesso dalla propria autorità di certificazione, ma deve necessariamente acquistare, pagandolo, un certificato X.509 generato da *VeriSign*, la quale si farà carico della verifica del rispetto delle regole di esportazione imposte dal governo statunitense.

Il formato PEM

Il formato PEM

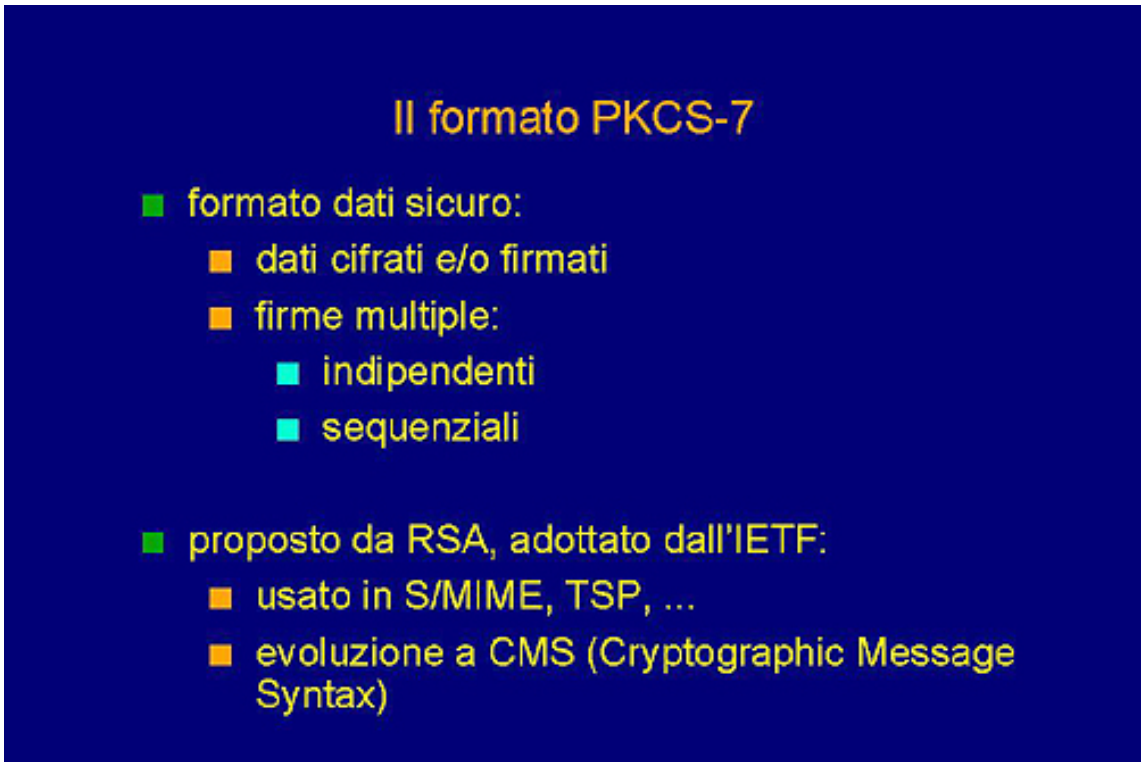
In generale, quando si parla di firma digitale si sta parlando di qualcosa di astratto. Se vogliamo applicare la firma digitale a dei documenti elettronici, quindi a dei *file*, tipicamente bisogna anche definire il formato con cui la firma digitale viene introdotta all'interno. Esistono vari formati, il primo definito nel passato è stato il formato PEM, che è stato sviluppato per rendere sicuri i messaggi di posta elettronica, ossia per generare messaggi firmati oppure messaggi con firma digitale ed anche cifratura, ossia segretezza. Questo è un formato molto vecchio, attualmente non viene più usato come formato dati, per trasmettere messaggi, ma solamente per salvare e proteggere sui dischi locali le chiavi pubbliche e private di alcuni processi. Ad esempio, il suo tipico utilizzo è all'interno dei *server Web* sicuri.

Il formato PGP

Il formato PGP

Un formato alternativo è il cosiddetto formato PGP. PGP è un programma che permette di fare crittografia e firma digitale sia dei messaggi di posta elettronica sia di generici *file*. È un programma molto utilizzato da persone non strutturate: come gli *hacker* o le persone appartenenti all'*underground* di *Internet* o semplicemente coloro che hanno particolarmente a cuore la *privacy* dei propri dati. Il formato PGP ha delle funzionalità analoghe a quelle di PEM, ossia è in grado di generare dei dati firmati ed eventualmente cifrati. PGP è passato da una versione *public domain* a una versione sviluppata da una ditta. Nella prima versione gli algoritmi utilizzati erano IDEA, RSA, MD5. Nella nuova versione, che è incompatibile con la precedente, viene utilizzato l'algoritmo 3DES, lo standard di firma digitale DSS e lo scambio chiavi D-H. Uno dei maggiori problemi che si frappongono all'adozione su larga scala del formato e dei sistemi di sicurezza basati su PGP, è il fatto che non utilizza certificati X.509 e questo causa difficoltà nella distribuzione delle chiavi pubbliche usate da PGP.

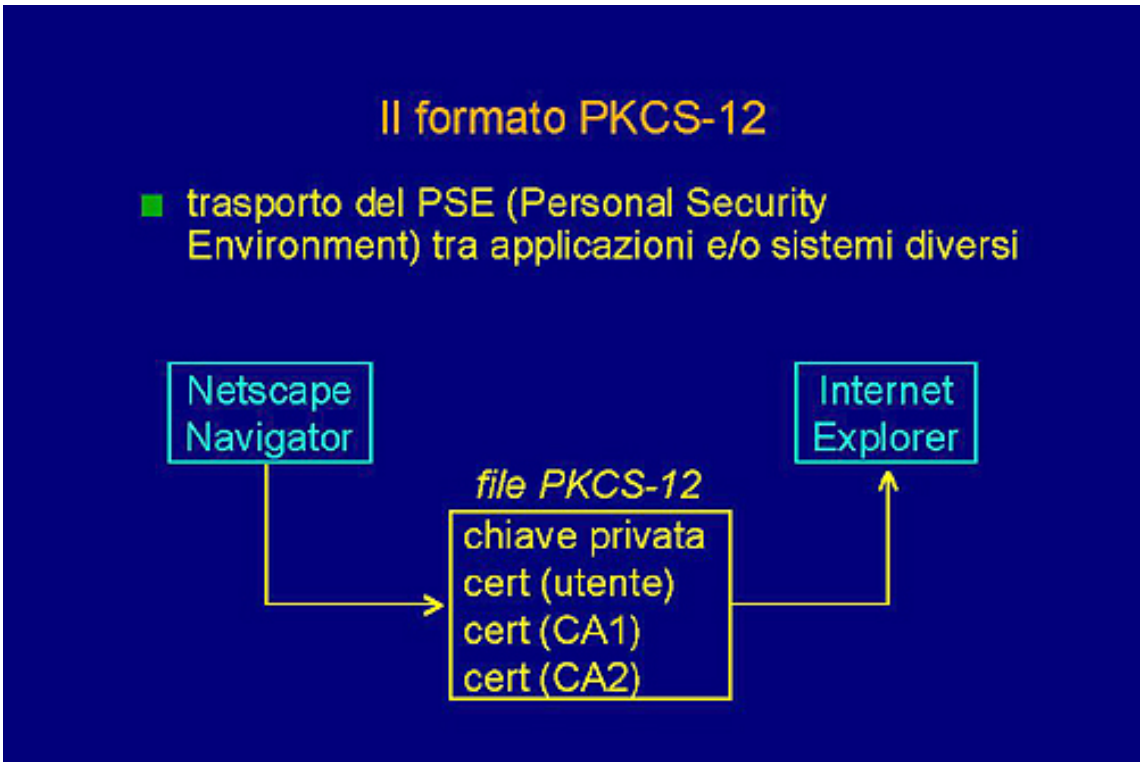
Il formato PKCS-7



Il formato PKCS-7

Attualmente il formato dati più utilizzato per apporre firme digitali o per cifrare i dati è il cosiddetto formato PKCS-7. È un formato molto flessibile che permette non soltanto di cifrare o firmare in modo semplice i dati, ma anche di apporre firme multiple, ossia il medesimo dato, il medesimo messaggio, può contenere le firme di più persone. Queste persone possono aver firmato il messaggio ciascuna in modo indipendente dall'altro, oppure possono averlo fatto sequenzialmente, ossia in pratica avendo visto chi aveva già firmato il messaggio in precedenza. Il formato PKCS-7 è stato sviluppato e proposto dalla ditta RSA, ma è stato recentemente adottato da *Internet*, ossia dall'IETF, l'ente che sviluppa tecnicamente le procedure per *Internet*. In particolare il formato PKCS-7 è usato nella posta elettronica sicura S/MIME nei protocolli per fornire ora e data certa. Il formato PKCS-7 sta evolvendo alla sintassi chiamata CMS (*Cryptographic Message Syntax*), che dovrebbe diventare la sintassi standard per la firma digitale e la segretezza dei documenti nei prossimi anni.

Il formato PKCS-12



Il formato PKCS-12

Infine, nel caso che noi abbiamo ottenuto una chiave privata e una chiave pubblica per utilizzo all'interno di una particolare applicazione o su un determinato calcolatore, si potrebbe porre il problema di come utilizzare questa nostra identità digitale anche su altri sistemi. In questo caso ci viene in aiuto il formato cosiddetto PKCS-12. PKCS-12 è un formato dati che è stato definito per permettere il trasporto del PSE, ossia del mio ambiente di sicurezza personale, fra applicazioni o sistemi diversi. Questo vuol dire, ad esempio, che se io utilizzo le funzionalità di firma digitale in *Netscape*, posso usare le medesime funzionalità anche in *Explorer* semplicemente esportando il mio ambiente di sicurezza in un *file* PKCS-12 protetto. Metto qua dentro, ad esempio, la mia chiave privata, il mio certificato di utente e i certificati delle CA che hanno emesso il mio certificato d'utente. Il tutto è poi fornito in pasto al sistema *Explorer*, che assumerà quindi la medesima identità digitale già disponibile sul mio sistema *Netscape*.

Bibliografia

Introduzione

Libri

B. Cooper*Javascript;* 2001Apogeo

S. Isaacs*Inside Dynamic HTML;* 1998Mondadori informatica

Marco Calvo, Gino Roncaglia, Fabio Ciotti, Marco A. Zela*Internet 2000;* 2000Editori Laterza

Siti

Guida HTML; http://html.it/

Dynamic HTML Overview;
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/dhtml_node_entry.asp

Sito della Adobe; http://www.adobe.com/main.html

Sito di FrontPage;
http://office.microsoft.com/assistance/topcategory.aspx?TopLevelCat=CH79001803&CTT=6&Orig

Glossario

ACL : (*Access Control List*). Abbreviazione per *Access Control List*, ovvero *Access List*. Nella tecnologia delle reti *Microsoft Windows* rappresenta l'elenco delle regole di accesso ad una risorsa (esempio cartella del *file system*) e delle restrizioni attive su tale risorsa. Nell'ambito del *networking* il termine indica un filtro sul traffico che un *router* o un ***firewall*** effettua a scopo di protezione di una rete, di un'**applicazione**, di una macchina.

ActiveX : Una tecnologia che si prefigge gli stessi scopi di ***Java*** ma non ad architettura aperta (è di proprietà della *Microsoft* e permette la realizzazione di moduli di codice incorporabili tramite OLE).

Applet : Un piccolo programma che può essere prelevato velocemente dalla rete e usato da qualsiasi *computer* dotato di un *browser* capace di eseguire codice ***Java***.

Applicazione : Un programma (*software*) che svolge determinate funzioni per l'utente finale. Esempi di applicazioni sono i *client* FTP, Telnet, *E-mail* e i *browser*.

DHTML : (*Dynamic* HTML). È una evoluzione di HTML per rendere meno

statico il codice delle pagine.

Dreamweaver : Programma sviluppato dalla *Macromedia* per la creazione di siti mediante un *editor*, funzioni di gestione che consentono di sviluppare siti *Web* a livello professionale.

Firewall : Dispositivo comprendente componenti *hardware* e *software* preposto al filtraggio di pacchetti a scopo di protezione di una rete, di specifiche macchine, di specifiche applicazioni.

FrontPage : Programma sviluppato dalla *Microsoft* per la creazione di siti mediante un *editor*, funzioni di gestione che consentono di sviluppare siti *Web* a livello professionale con immagini, suoni e animazioni.

GARR : (Gruppo Armonizzazione Reti per la Ricerca). Organismo patrocinato dal MURST (Ministero per l'Università e la Ricerca Scientifica e Tecnologica) per la gestione e lo sviluppo della rete omonima.

GoLive : Programma sviluppato dalla *Adobe* per la creazione di siti mediante un *editor*, funzioni di gestione che consentono di sviluppare siti *Web* a livello professionale con immagini, suoni e animazioni.

ISO : (*International Standard Organization*). Principale organismo di standardizzazione mondiale di cui fanno parte gli organismi di standardizzazione nazionali quali l'ANSI per gli USA e l'UNINFO per l'Italia.

ISP : (*Internet Service Provider*). Fornitore di servizio di accesso ad *Internet*. Generalmente, per gli utenti residenziali l'accesso è fornito mediante collegamento telefonico al POP del *provider*, mentre per categorie di utenti di tipo affari, il collegamento può essere su linea dedicata e collegamento diretto numerico fra la sede dell'utente ed il *router* del *provider*.

JAVA : Un linguaggio di programmazione *Object Oriented*, sviluppato da *Sun Microsystems* e disponibile già da diversi anni, specificatamente progettato per la scrittura di programmi che possono essere scaricati sul proprio *computer* dalla rete ed immediatamente eseguiti localmente. Utilizzando piccoli programmi ***Java*** (chiamati *Applet*), le pagine *Web* possono includere animazioni, effettuare calcoli e quant'altro.

Javascript : Mentre un programma scritto in ***Java*** va sottoposto ad un processo di meta-compilazione per poter essere eseguito, ***Javascript*** è un linguaggio interpretato che può essere inserito direttamente nel codice HTML dei documenti *Web*.

Linguaggio di *Scripting* : Un termine generico per qualunque linguaggio che è debolmente tipato o senza tipi, e non consente di utilizzare strutture dati complesse. Un programma in questo linguaggio è di solito interpretato.

LOG (*file* di) : Sono *file* che contengono informazioni sugli accessi ad un sistema, come indirizzo della macchina remota, ora di accesso, eccetera.

OSI : (*Open Systems Interconnection*). Standard internazionale, dell'**ISO**, descritto nel documento **ISO** 7498, per un modello di riferimento per l'interconnessione di sistemi; è organizzato in 7 livelli (*Physical*, *Data Link*, *Network*, *Transport*, *Session*, *Presentation*, *Application*), ciascuno dei quali si basa sui servizi forniti dal sottostante strato e fornisce a sua volta servizi allo strato sovrastante. Lo scopo è di realizzare sistemi aperti, capaci di far comunicare sistemi diversi fra loro.

Plug In : Componente *software* integrabile in un *browser* allo scopo di visualizzare (in senso lato) contenuti multimediali espressi in formato non HTML.

POP : (*Post Office Protocol*). È il protocollo utilizzato dal *client* di posta elettronica per richiedere al ***server*** associato i messaggi ricevuti.

Router : Dispositivo fisico operante a livello 3 del modello OSI, in grado di effettuare il *forward* dei pacchetti in base alle regole su cui si basano i livelli 3 delle reti a cui risulta connesso e per le quali svolge il servizio di *routing*.

Script : Un programma scritto in un **linguaggio di *scripting***.

Sincrona : Tipo di trasmissione dati in cui la comunicazione tra trasmettitore e ricevitore avviene allo stesso tempo, o alla stessa velocità, o in modo regolare e predicibile.

Server : Un *computer* o un programma che fornisce un determinato tipo di servizio ad un programma *client* in esecuzione su un *computer* remoto. Una stessa macchina può eseguire contemporaneamente più di un programma fungendo quindi da più ***server*** per molti *client* sulla rete.

Servlet : Un programma per ***server*** che garantisce funzionalità aggiuntive ai ***server*** abilitati ***Java***.

Tag : Marcatore, racchiuso tra parentesi angolari, che costituisce l'elemento caratterizzante l'HTML (per esempio: <h1>, </H1>, <IMG SRC="stella.gif">).

Telnet : Applicazione *Internet* basata sul protocollo TCP, che consente di remotizzare attraverso una rete IP, l'accesso a un *host* remoto. Il *client* si comporta come terminale remoto del ***server* telnet** ed accede alle risorse dell'*host* mediante l'autenticazione con una *username* ed una *password*.

Variabile : Un elemento di dati indicato da un identificatore. Ogni **variabile** ha un tipo, ad esempio un numero intero od oggetto e un ambito.

Autori

Hanno realizzato il materiale di questo modulo:

Prof. Cosimo Laneve

Professore Straordinario di Informatica presso l'Università di Bologna, dove insegna Linguaggi di Programmazione e Qualità del *Software*. Ha ricevuto il Dottorato di Ricerca in Informatica dall'Università di Pisa ed è stato *Research Associate* presso l'INRIA di *Sophia Antipolis* in Francia. Attualmente è coordinatore di progetti nazionali e internazionali che riguardano i fondamenti teorici e l'implementazione di linguaggi di programmazione distribuiti, di verifica statica di programmi e di teoria dei tipi. Relativamente a queste tematiche, ha pubblicato su numerose riviste e conferenze internazionali.