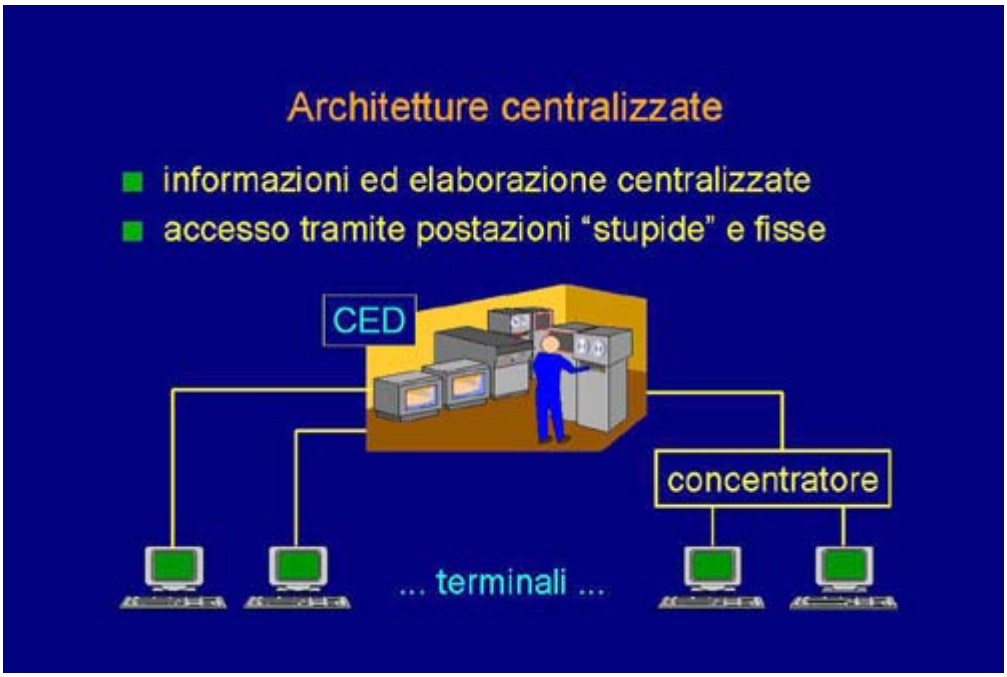
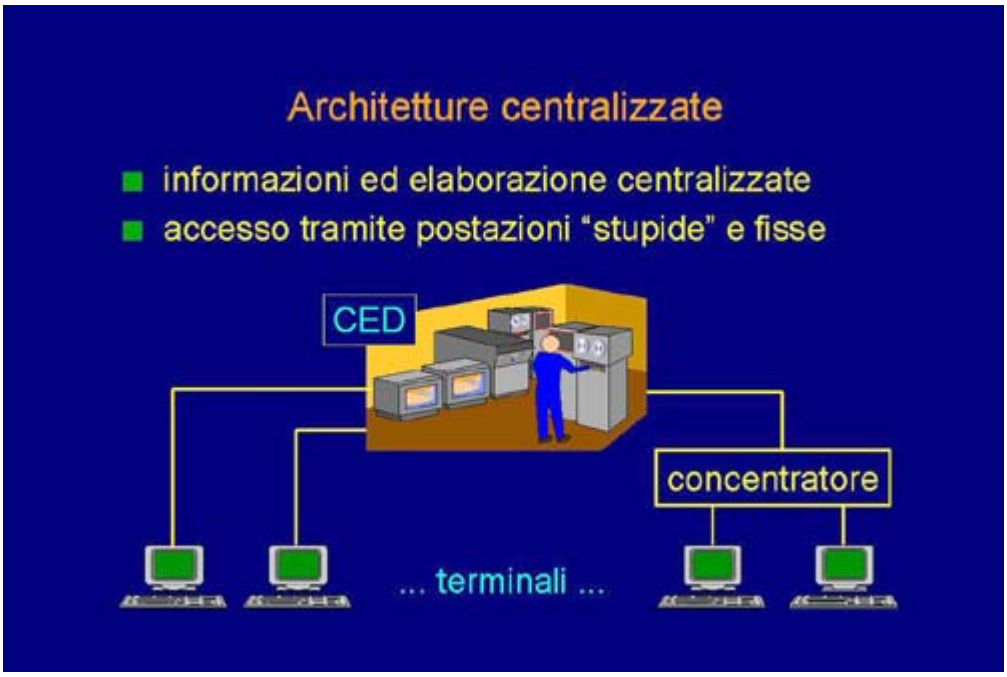
Architetture dei Sistemi Informativi Distribuiti
Architetture centralizzate (1/2)



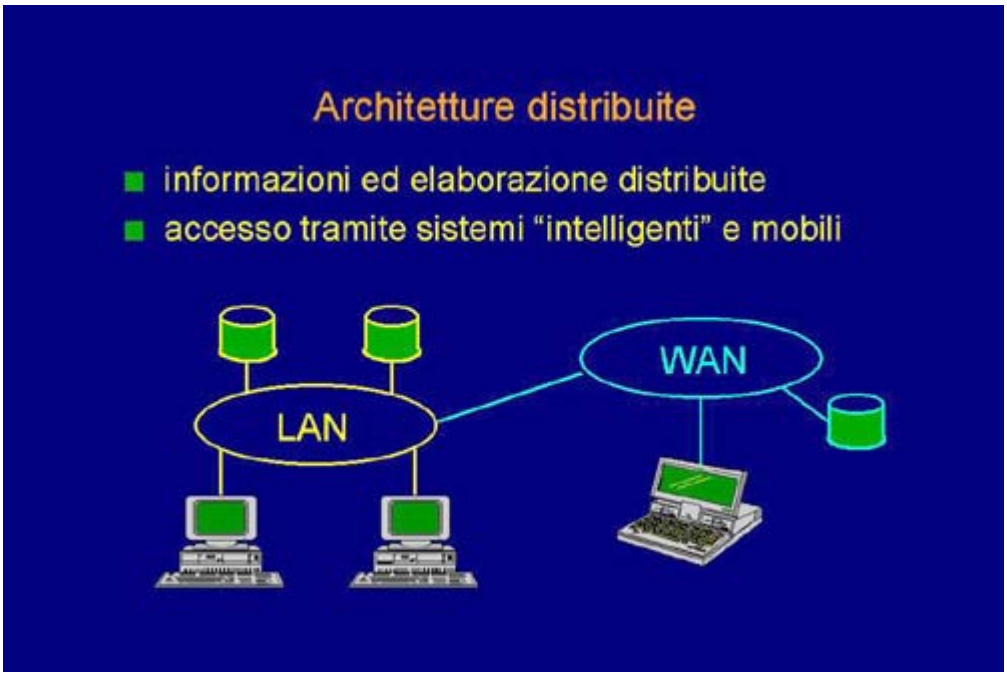
Con questa lezione iniziamo la trattazione relativa alle architetture dei sistemi informativi di tipo distribuito: questi sono i sistemi informativi che oggi vengono realizzati nella maggior parte dei casi, ma per capirne appieno le potenzialità, le funzionalità e l'architettura, conviene prima fare un passo indietro ed esaminare quelli che erano i sistemi informativi di tipo centralizzato che regnavano sovrani fino a circa dieci anni fa e che ancora oggi in molti casi esistono e vengono utilizzati. In un'architettura centralizzata ritroviamo dei calcolatori e delle unità di memoria che sono tipicamente concentrati in un'unica stanza, normalmente denominata CED (Centro Elaborazione Dati) oppure SED (Servizio Elaborazione Dati); quindi in un sistema centralizzato tutta la potenza di calcolo e tutta la potenza di memoria sono memorizzate fisicamente in un unico luogo, chiuso, accessibile soltanto agli operatori. Gli utenti possono interagire con il sistema solo ed esclusivamente attraverso i terminali, questi ultimi sono a tutti gli effetti delle postazioni di lavoro fisse e, per così dire, stupide, nel senso che permettono solo ed esclusivamente di fare delle operazioni di input-output, dato che un classico terminale non è dotato altro che di una tastiera ed un video che permettono di fornire dati al sistema di elaborazione collocato all'interno del CED e di prendere le risposte e visualizzarle sul video della postazione di lavoro.

Architetture centralizzate (2/2)



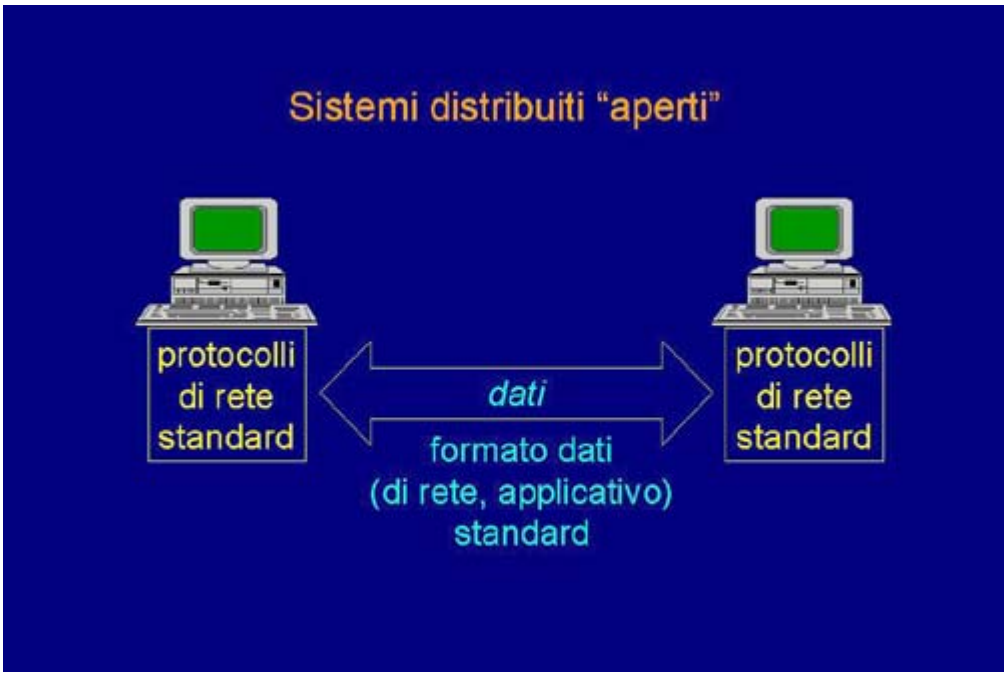
Un terminale quindi non possiede né una sua CPU né una sua capacità autonoma di memorizzare dei dati. I terminali vengono normalmente collegati ai sistemi di elaborazione tramite delle linee dedicate, nel senso di linee seriali, cavi coassiali, che corrono dalla postazione di lavoro dell'utente fino al tipico scantinato, in cui erano posizionati i grossi sistemi di calcolo. Nel caso che la distanza fra il terminale e il nodo di elaborazione sia eccessiva, tipicamente i terminali non venivano collegati direttamente al nodo di elaborazione, ma, come mostrato qui in figura, venivano collegati prima ad un concentratore, cioè un'apparecchiatura di telecomunicazione che si faceva carico di concentrare più linee e di convogliare il flusso di input-output su un'unica linea dedicata, ad esempio un collegamento via modem fra una sede remota e il nodo di elaborazione. Notate quindi che queste postazioni sono stupide e fisse e notate altrettanto bene che non è la stessa cosa dire che al posto di un vero e proprio terminale si usa un personal computer in modalità emulazione di terminale, perché il personal computer è invece dotato di una sua autonoma capacità di elaborazione e di una sua autonoma capacità di memorizzazione dei dati. L'avvento dei personal computer e delle reti sono i due elementi base che hanno portato alla morte dei vecchi sistemi informativi centralizzati e alla nascita e all'avvento dei sistemi informativi di tipo distribuito.

Architetture distribuite



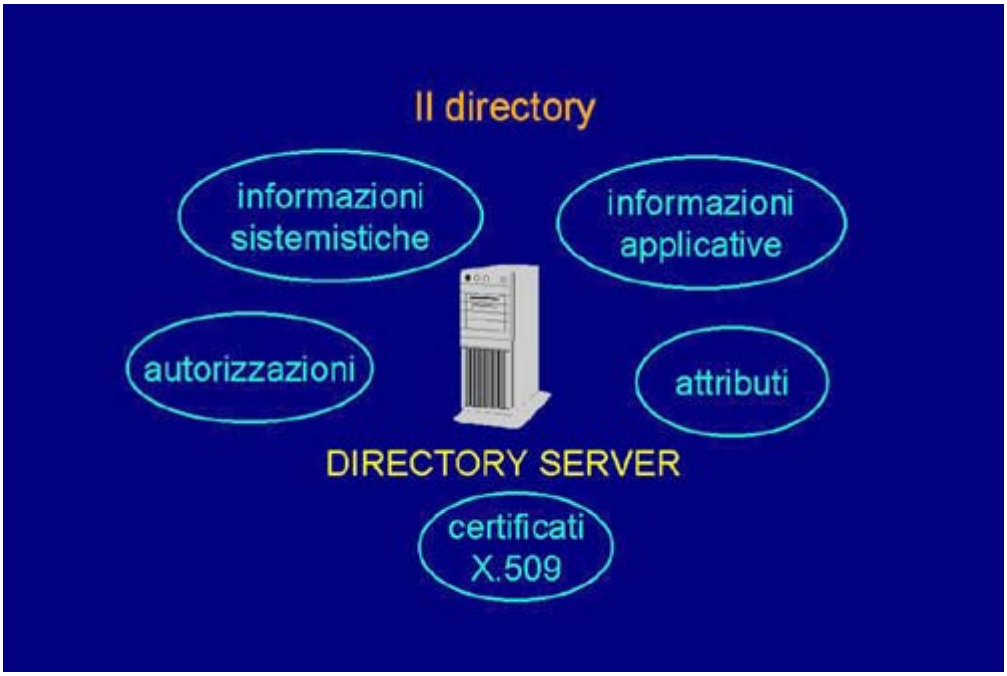
Un sistema informativo di tipo distribuito può essere rappresentato secondo lo schema che vedete qui indicato: le postazioni di lavoro sono diventate dei nodi autonomi di elaborazione: sono i personal computer, sia di tipo fisso, cioè quelli che troviamo sulle scrivanie dei nostri tavoli di lavoro, ma anche di tipo mobile, cioè quelli usati dalle persone che per vari motivi devono viaggiare molto e quindi hanno necessità di portarsi appresso la propria capacità di elaborazione. Quindi, pur continuando ad esistere dei server, che sono attaccati alla rete locale o alla rete geografica, non è più vero che i server sono gli unici depositari della potenza di calcolo e della capacità di memorizzazione. È ancora vero che c'è della capacità di memorizzazione, ma questa è tipicamente distribuita. In sostanza, il grosso cambiamento sta nel fatto che sia la capacità di elaborazione che la capacità di memoria non sono più concentrate in un unico luogo, ma distribuite; inoltre possono essere mobili e possono essere distribuite tra i responsabili del servizio, cioè quelli che gestiscono il successore del CED, il server di elaborazione, ma distribuite in parte anche tra gli utenti stessi che possono autonomamente svolgere delle operazioni e memorizzare dei dati. Cerchiamo quindi di capire quali sono gli elementi base da un punto di vista software che caratterizzano un'architettura di tipo distribuito.

Sistemi distribuiti aperti



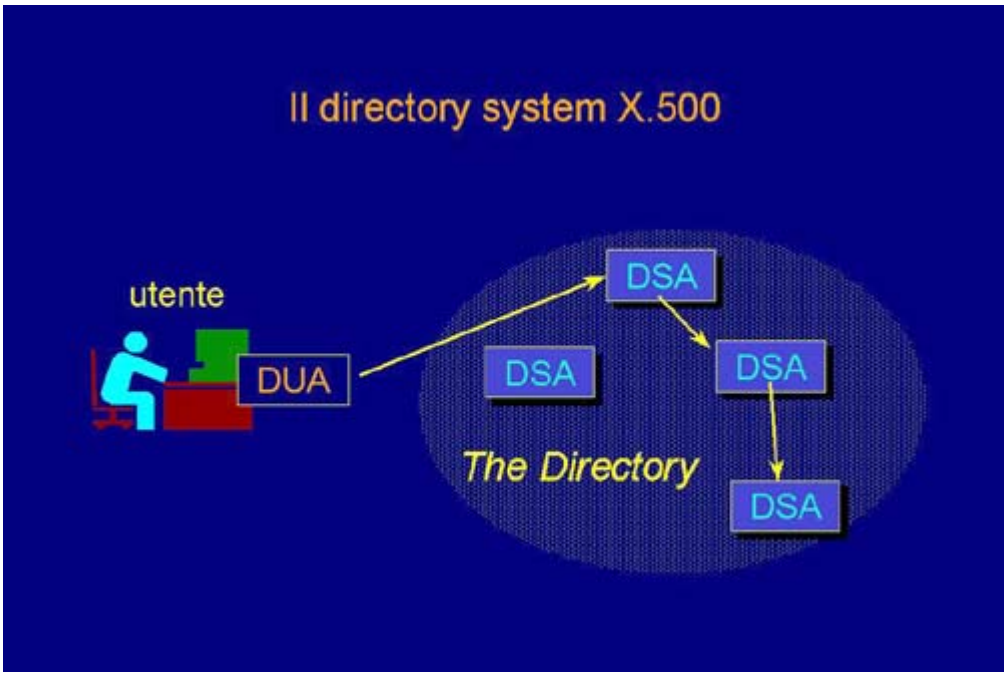
Innanzitutto affinché si possa parlare di un sistema distribuito di tipo aperto occorre che i nodi di elaborazione che desiderano comunicare, debbano essere dotati di protocolli di rete standard; notate che per comunicare devono essere dotati di protocolli di rete, in particolare poi per comunicare in modo aperto, ossia senza restrizione su quali tipi di nodi possono comunicare, bisogna implementare degli standard aperti, quali ad esempio gli standard della rete TCP/IP. Se i due nodi di elaborazione implementano questi protocolli di rete standard, ecco allora che possono scambiare dei dati. È altrettanto importante che anche il formato di questi dati, sia a livello di pacchetto di rete che di tipo applicativo, sia altrettanto standard, perché questa è l'unica garanzia che abbiamo di realizzare delle applicazioni che non siano vincolate ad un unico fornitore, ad un'unica particolare architettura hardware o software. Quindi la parola standard è da intendersi sia a livello di protocolli di telecomunicazione, sia a livello di formato dei dati grezzi o del formato dei dati applicativi.

Il directory



In particolare, oggi come oggi, uno degli elementi che si stanno imponendo all'attenzione di tutti i progettisti di sistemi distribuiti aperti per la sua capacità di creare, di memorizzare informazioni e distribuirle all'interno della rete è il cosiddetto directory. Il directory viene normalmente implementato tramite un server ad esso dedicato che mantiene una serie di informazioni e le fornisce a tutti i nodi della rete. Questo ci permette di avere in un unico posto le informazioni che servono e far sì che queste vengano distribuite a tutti i nodi; in particolare il directory è emerso come un elemento essenziale quando si è cominciato a pensare di voler fare sicurezza di un sistema distribuito, perché tramite il directory noi possiamo fornire i cosiddetti certificati a chiave pubblica X.509 che sono uno degli elementi fondamentali, portanti, di tutte le architetture di sicurezza per sistemi distribuiti. Ma questo è solo uno dei tipi di dati che si possono memorizzare nel directory; ad esempio il directory può essere utilizzato per memorizzarci informazioni sistemistiche, quindi i nomi degli utenti, le password, i privilegi di ciascun utente. Oltre a questo tipo di informazioni sistemistiche, una volta che viene messo in piedi un directory server, viene ovviamente voglia di convogliare su di esso più informazioni possibili per semplificarne la gestione e per evitare di avere copie di dati non allineate sui vari sistemi. Quindi oltre alle informazioni sistemistiche si possono inserire anche informazioni di tipo applicativo, ossia che dipendono dalla particolare applicazione che un utente desidera usare. Si possono inserire anche degli attributi non necessariamente correlati al sistema operativo, ma correlati alle funzioni applicative che un utente svolge. Si possono mettere anche le autorizzazioni, che sono tipicamente il tramite tra gli utenti e le applicazioni, ossia un'autorizzazione non è niente altro che la concessione o la negazione del permesso per un certo utente di accedere a determinate risorse di tipo informatico.

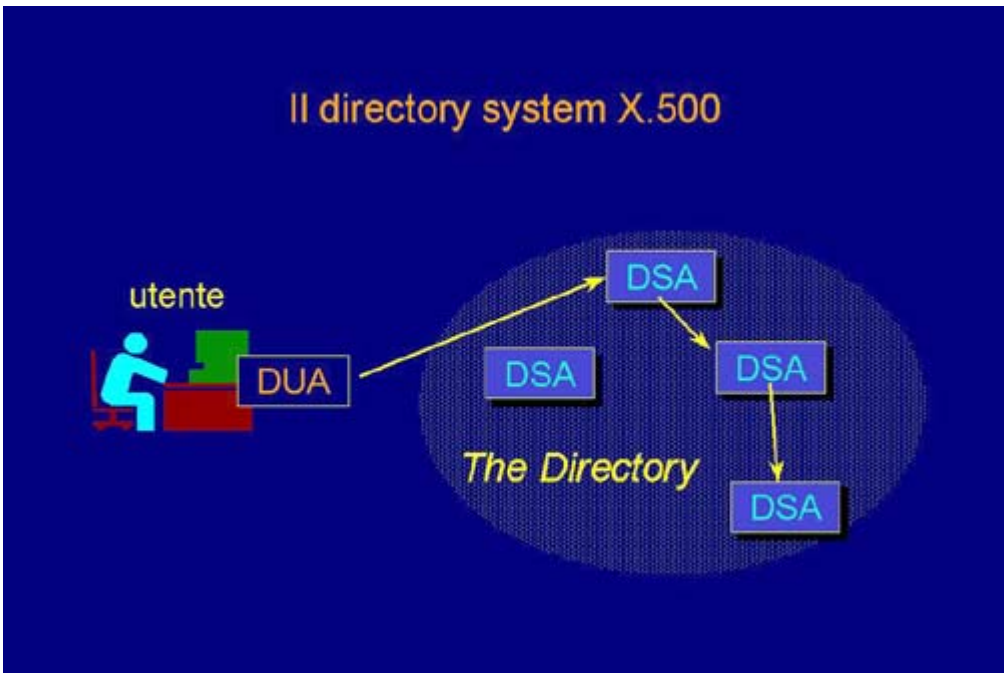
Il directory system X.500 (1/2)



Esistono due modelli base per realizzare un directory. Esso è nato molti anni fa e si è imposto all'attenzione informatica con la rete OSI; all'interno di questa rete era stato sviluppato un servizio applicativo chiamato X.500 perché scritto nello standard ITU X.500. Il directory system X.500 è schematizzabile nel seguente modo: l'utente che desidera interagire con il directory system deve essere dotato di un opportuno client chiamato DUA (Directory User Agent). Tramite il Directory User Agent l'utente può interrogare questa grossa nuvola che contiene i dati. Questi ultimi vengono chiamati genericamente the Directory e sono gestiti da singoli server; nella terminologia X.500 i server vengono a sua volta chiamati DSA (Directory System Agent). Ogni Directory System Agent mantiene una certa quantità di informazioni, ad esempio potremmo pensare che ogni Directory System Agent gestisca le informazioni relative ad un certo dipartimento all'interno di

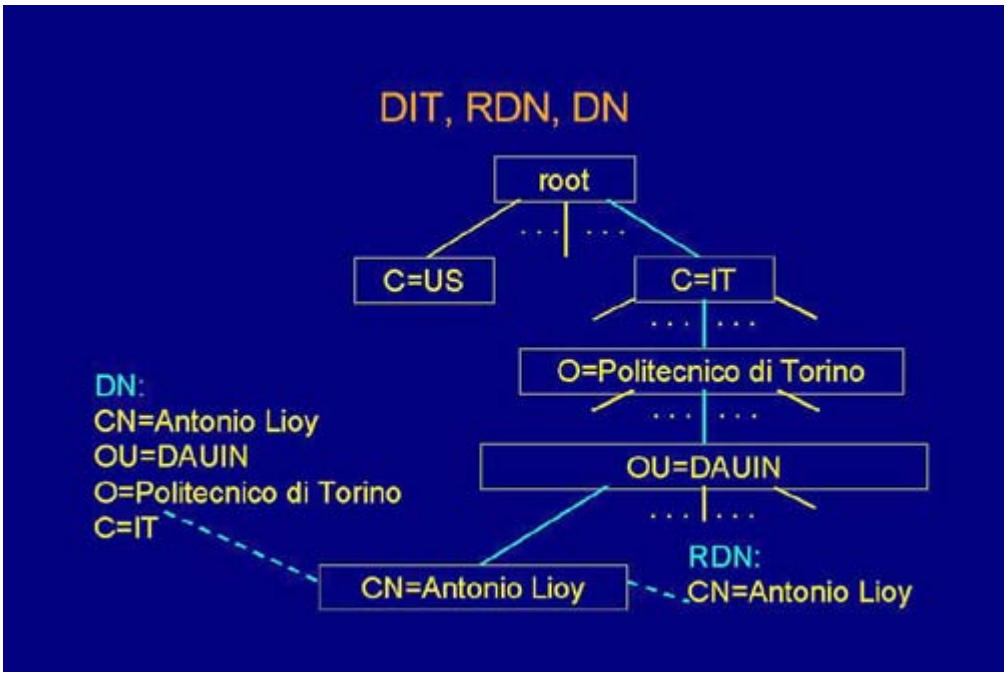
un'organizzazione, oppure nel caso di un'organizzazione soprannazionale potremmo pensare che ogni directory gestisca le informazioni relative ad un certo paese. Ogni directory ha delle informazioni che altri non hanno, quindi nel momento in cui il nostro utente fa riferimento ad un Directory System Agent specifico può darsi che trovi le informazioni che cerca, come può darsi che non le trovi. Nel caso in cui l'utente non trovi le informazioni nel primo Directory System Agent lo standard X.500 prevede che i vari Directory System Agent possano parlarsi tra di loro tramite l'operazione cosiddetta di chaining, cioè concatenazione, con questa operazione i vari Directory System Agent fanno automaticamente la ricerca richiesta dall'utente e sono in grado di fornirgli la risposta.

Il directory system X.500 (2/2)



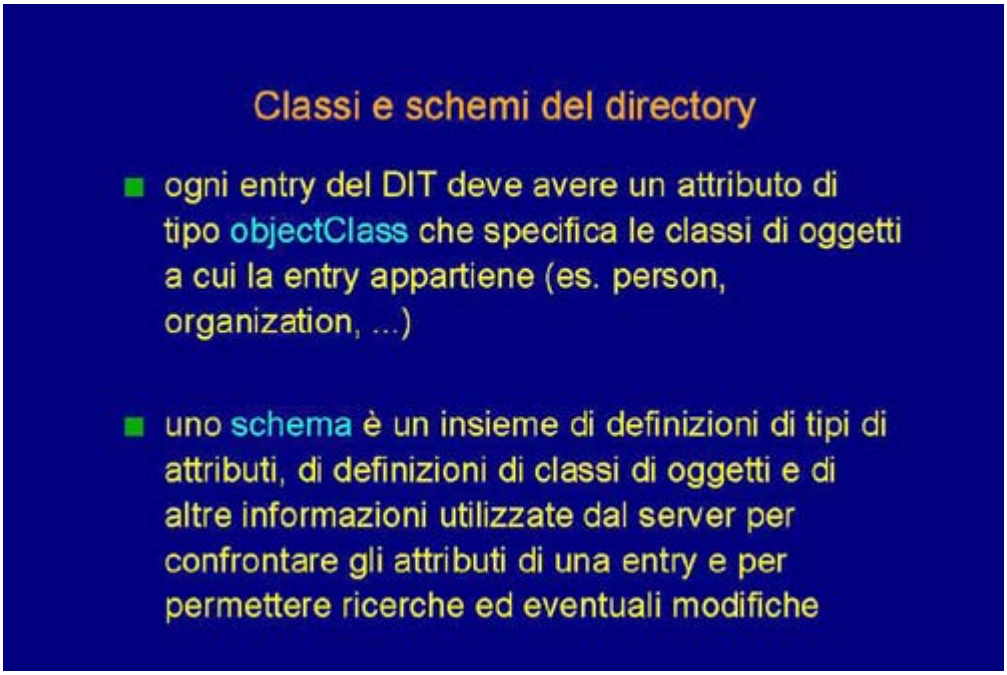
Uno dei più grossi problemi esistenti nel directory system X.500 è che il meccanismo di chaining è definito a livello di formato, di scambio di dati, ma non è definito a livello organizzativo, ossia la ricerca dell'informazione segue un cammino di amicizia: il gestore di un certo server è a conoscenza dell'esistenza di un altro server e quindi gli passa le informazioni, ma non esiste un'organizzazione, né a livello nazionale né a livello mondiale, che permetta con certezza di trovare tutte le informazioni esistenti. Il directory system X.500 teoricamente sarebbe in grado di creare un directory a livello mondiale, in realtà purtroppo è frammentato all'interno di tante piccole realtà. X.500, facendo parte dello standard OSI, ha seguito un po' il fatto di questo standard che è declinato dopo la sua apparizione e gli entusiasmi che inizialmente aveva destato, per la sua complessità di implementazione. Oggigiorno sappiamo tutti quanti che le reti che vanno per la maggiore non sono quelle standard OSI, ma sono quelle standard TCP/IP. Nell'ambito di questo TCP/IP il directory è stato rivisitato, vedremo tra un attimo come.

DIT, RDN, DN



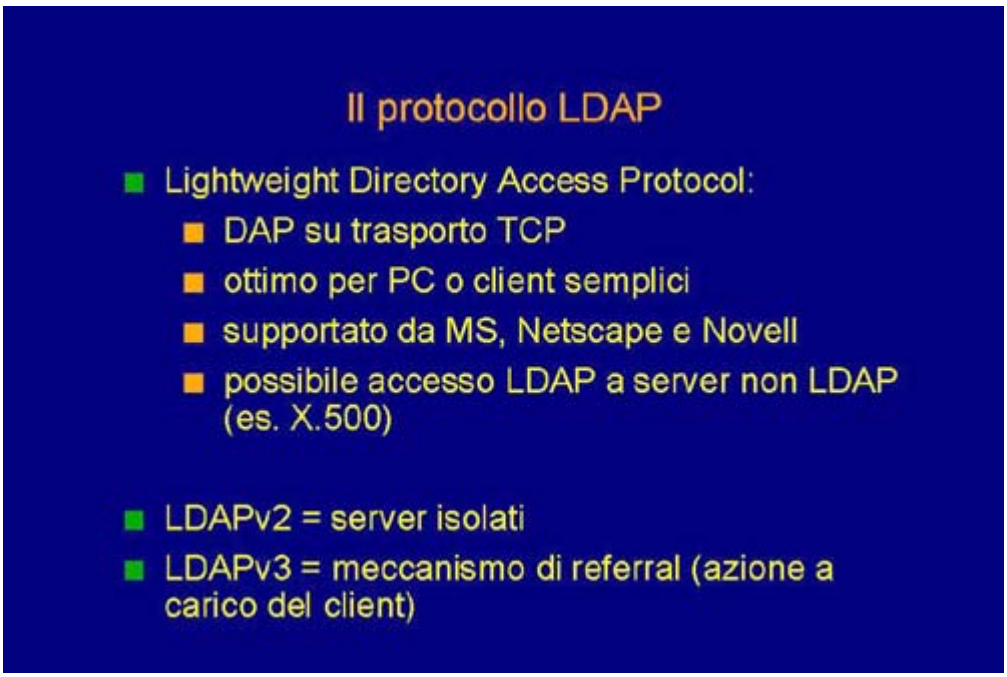
Prima intendo sottolineare che qualunque sia il tipo di directory che noi adottiamo, sia il modello X.500, sia il modello di TCP/IP, i dati dentro il directory vengono organizzati gerarchicamente, ciò vuol dire che esistono vari nodi, ognuno dei quali corrisponde al cosiddetto distinguished name di tipo relativo. Ad esempio, al primo livello i nodi identificano i paesi, indicati con C (Country), in questo caso Italia; al secondo livello i nodi identificano le organizzazioni: qui sto seguendo la strada che mi porta al Politecnico di Torino; e, al terzo livello si identificano tipicamente le unità organizzative, in questo caso il mio dipartimento di Automatica ed Informatica. Sotto questo livello esistono ancora degli altri nodi che identificano le persone,o gli oggetti, l'entità di rete, tramite il loro nome comune. La strada che viene seguita a partire dalla radice, dalla base del directory, per arrivare all'informazione cercata, quindi la sequenza dei RDN (Relative Distinguished Name) costituisce il cosiddetto DN (Distinguished Name), il nome distinto, unico di quell'oggetto; in questo caso questa è la sequenza che costituisce il mio Distinguished Name.

Classi e schemi del directory



Per ognuno dei dati che vengono messi in questo albero, detto DIT (Directory Information Tree) vengono definiti degli attributi che sono specificati come una classe, ossia ogni entry nel directory appartiene ad una classe di oggetti per la quale sono stati predefiniti degli attributi, ossia dei dati che lei stessa può avere. Poiché uno stesso directory può contenere oggetti di tantissime classi, ecco che è necessario definire quello che viene indicato con schema, cioè l'insieme di tutti quanti i dati che possono risiedere nel directory; vedete nella slide che ho indicato lo schema come un insieme di tipi, attributi, classi ed altre informazioni genericamente utilizzate dal server per effettuare ricerche o modifiche dei dati memorizzati. Come vi dicevo, all'interno delle reti TCP/IP non si è seguito lo standard X.500 puro, ma si è seguita una strada X.500 modificata, semplificata, per renderla effettivamente applicabile.

Il protocollo LDAP



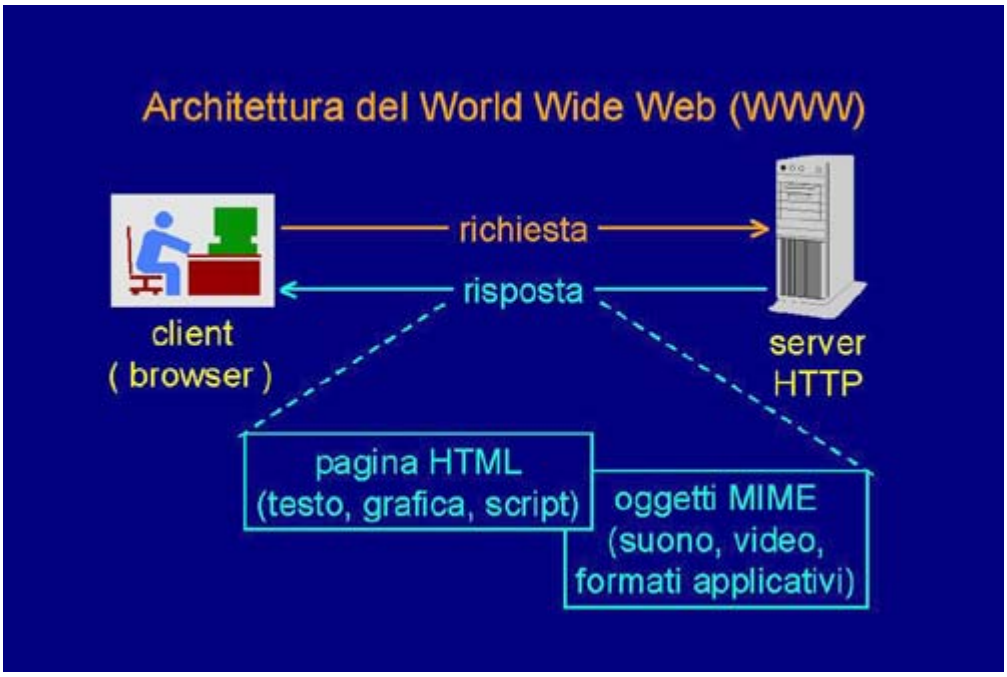
Questa strada ha portato alla definizione di un nuovo protocollo per accedere ai dati, chiamato LDAP ed è un protocollo standard nelle reti TCP/IP per accedere dal client ai server di directory. LDAP significa protocollo leggero per l'accesso al directory, ossia non è altro che il protocollo DAP, originariamente usato da X.500, portato su un trasporto, su un canale di tipo TCP. È bastata questa semplice sostituzione per rendere questo protocollo ottimo per applicazioni anche su client molto semplici quali personal computer o addirittura dispositivi portatili. Il grande successo di LDAP è stato sancito da una dichiarazione pubblica del Settembre 1997 in cui Microsoft e Netscape hanno dichiarato che era loro intenzione convergere sul sistema LDAP come sistema per distribuire le informazioni all'interno delle loro applicazioni. Notate che per utilizzare LDAP non è indispensabile avere un server LDAP, esistono infatti anche dei server X.500 che sono in grado di parlare entrambi i protocolli, ossia sono in grado di parlare DAP verso i Directory User Agent e di parlare LDAP verso i client più semplici. Di LDAP esistono tre versioni, a parte la prima versione che ha ormai un interesse più storico, oggigiorno le due versioni più utilizzate sono LDAPv2 e LDAPv3. La differenza fondamentale tra queste due implementazioni consiste nel fatto che in LDAPv2 ogni server di directory è un server completamente slegato da tutti gli altri; quindi se noi poniamo la nostra domanda ad un server LDAPv2 e questo non conosce la risposta, sarà compito nostro andarci a cercare in giro per il mondo un altro server che sia in grado di soddisfare la nostra domanda. In LDAPv3 invece è possibile che il server a cui abbiamo fatto la domanda sappia di non avere la risposta, ma nel contempo sappia indicarci lui stesso, tramite il meccanismo detto di referral, un altro server che potrebbe avere le informazioni che noi cerchiamo. In tal caso il server LDAPv3 si limita a fornirci un riferimento, ma l'azione di andare a chiedere la nuova informazione

sarà completamente a carico del client.

Uso di LDAP da parte delle applicazioni

LDAP oggi giorno è largamente utilizzato in tantissime applicazioni; in particolare si trovano dei client LDAP all'interno di tutti i sistemi più moderni di posta elettronica; in questo modo è possibile realizzare una rubrica globale al posto di una rubrica personale, ossia un elenco di nomi, cognomi, organizzazioni e indirizzi di posta elettronica comuni a tutte le persone che utilizzano quel directory; è anche possibile mettere all'interno del medesimo directory certificati a chiave pubblica X.509 che servono per fare sicurezza della posta elettronica. I server LDAP vengono utilizzati anche per fare funzioni di autenticazione e di autorizzazione: ad esempio ci sono molte apparecchiature di tipo NAS, ossia i server che fanno da front-end ai modem, quando da casa ci si collega ad Internet, che chiedono se si è autorizzati ad entrare ad un LDAP server. Questo ad esempio è estesamente utilizzato da tutti quei fornitori che forniscono servizi internet gratuiti, perché permette una fascia di integrazione anche con la gestione delle caselle di posta elettronica e delle home directory sul Web. Un'ultima parola riguardo il nuovo sistema operativo di Microsoft, Windows 2000, il quale ha rivoluzionato la propria architettura interna adottando un directory unico di tipo LDAP, dentro al quale va a memorizzare informazioni che prima erano suddivise fra il cosiddetto registry, tra vari file di sistema e include anche le informazioni che classicamente sono memorizzate nel DNS, ossia il sistema che permette di tradurre da nomi ad indirizzi. Quindi LDAP si sta decisamente affermando come il sistema per distribuire informazioni all'interno di un sistema di rete.

Architettura del World Wide Web

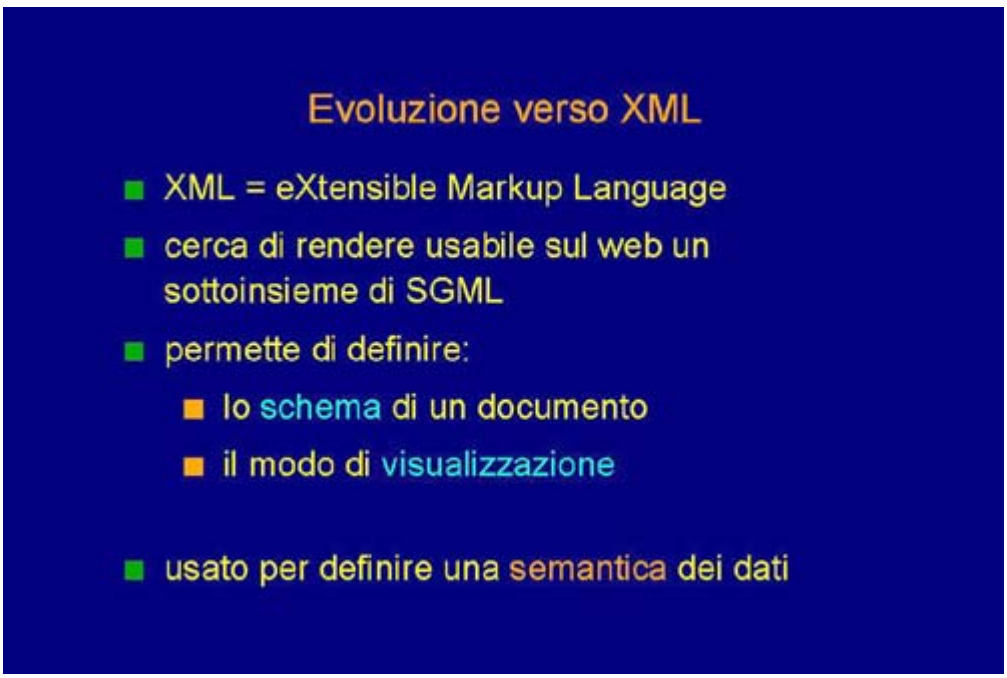


Se da una parte LDAP è uno degli elementi principali di un sistema distribuito, dall'altra parte a livello applicativo la tecnologia più utilizzata è quella del World Wide Web (WWW). Vediamo quindi un attimo quali sono i presupposti tecnici del World Wide Web. Il WWW si basa sull'esistenza di client chiamati browser, e di server; in particolare i client e i server devono utilizzare per parlarsi il protocollo HTTP. Tale protocollo prevede che il browser invii una richiesta verso il server, ad esempio la richiesta di visualizzare una certa pagina. Il server a questo punto fornisce la risposta alla domanda del client e tale risposta può contenere varie cose; di base contiene una pagina scritta in linguaggio HTML, che è un linguaggio ipertestuale ed ipermediale utilizzato dal Web. In questo modo nella pagina è possibile mettere del testo, della grafica ed anche degli script, ossia dei piccoli programmi che mi permettono di fare delle azioni un po' particolari. È anche possibile introdurre all'interno della pagina HTML degli oggetti cosiddetti MIME, cioè oggetti multimediali, come ad esempio del suono, del video o dei formati applicativi. Bisogna però sottolineare che questi oggetti MIME non fanno parte dei dati supportati in modo nativo dai client e dai server Web, ma devono essere trattati tramite i cosiddetti plug-in, ossia del software aggiuntivo da installare sul nostro browser.

Il linguaggio HTML

Ho detto che i dati devono essere espressi secondo il formato HTML, che vediamo indicato in questa slide di esempio; l'HTML è un linguaggio basato su tag, ossia su delle dichiarazioni, questa, <HTML>, ad esempio dice che qui inizia il file HTML, qui invece con <TITLE> inizia il titolo del documento e qui finisce, dove troviamo </TITLE>.In mezzo alle due troviamo il titolo vero e proprio. Al centro troviamo invece il corpo del documento, tra <BODY> e </BODY> che contiene non solo del testo, ma anche un link, ossia questa parola, AIPA, se premuta con il mouse mi porterà automaticamente verso questo sito Web, e contiene anche uno script, cioè un mini-programma scritto in un particolare linguaggio di programmazione, ad esempio Javascript, e qui ci saranno le informazioni relative a questo script. Quindi il linguaggio HTML mi permette di inserire all'interno del medesimo documento vari tipi di informazioni.

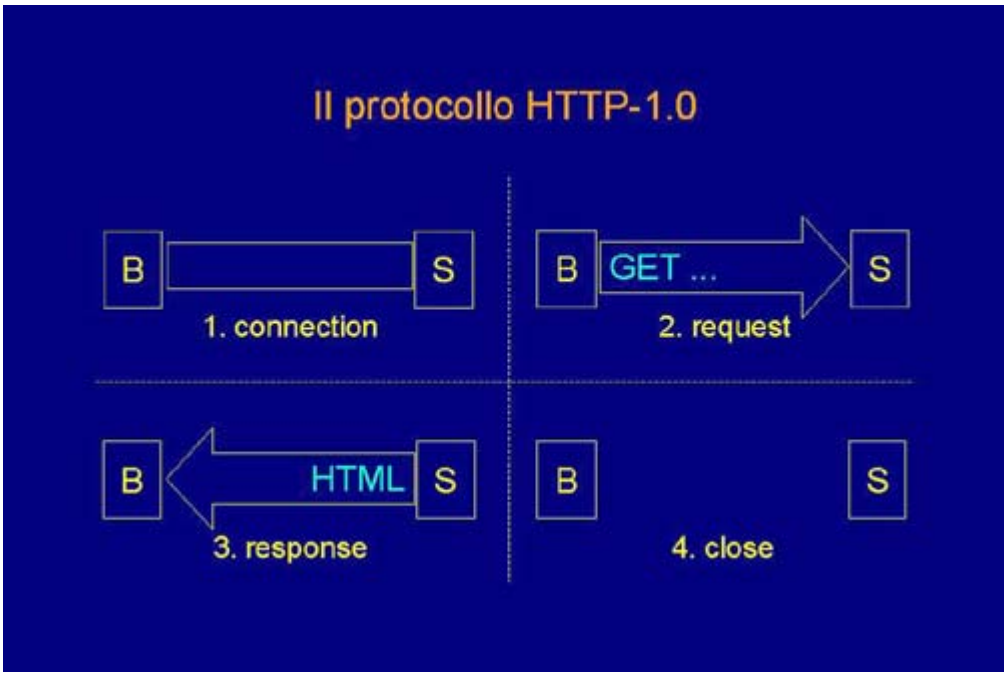
Evoluzione verso XML



HTML è stato il linguaggio che ha permesso la nascita del Web, ma ha un certo insieme di

restrizioni: permette di formattare i documenti per la visualizzazione, permette di formattarli un po' meno bene per la stampa, ma soprattutto non permette di attribuire una semantica ai dati che noi abbiamo inserito. A questo scopo è stato sviluppato un nuovo linguaggio che oggi sta destando molto interesse, chiamato XML, ed è una estensione dell'HTML, infatti il nome stesso XML significa eXtensible Markup Language. Tale linguaggio è stato sviluppato per cercare di rendere usabile, tramite il Web, un sottoinsieme di un linguaggio ancora più generale chiamato SGML, quest'ultimo è infatti il progenitore sia dell'HTML che del XML. In generale XML permette di definire non soltanto i dati che costituiscono il documento, ma lo schema a cui questi dati ubbidiscono e anche il modo in cui i dati devono essere visualizzati, quindi chi realizza il documento ha più controllo sul modo in cui deve essere visualizzato; ma soprattutto tramite XML è possibile definire una semantica dei dati, ossia è possibile non limitarsi semplicemente a dire che questo è un titolo, ma è possibile dire che questo è un titolo che rappresenta ad esempio il nome e cognome della persona fisica che ha realizzato questa pagina, quindi attribuire un significato alle parole e ai dati presenti dentro un file XML. Se questo da una parte arricchisce moltissimo il linguaggio, dall'altra parte presenta potenzialmente grandissime incompatibilità, perché ognuno di noi sarà in grado di definire un proprio schema di interpretazione dei dati XML, ma questo ovviamente contrasta con la necessità di standardizzare. Quindi XML rappresenta sicuramente un grosso passo in avanti nel rendere più ricco di contenuti informativi il Web, ma bisogna fare molta attenzione a come viene implementato perché si corre il rischio di creare delle nicchie che non interoperano in alcun modo tra di loro.

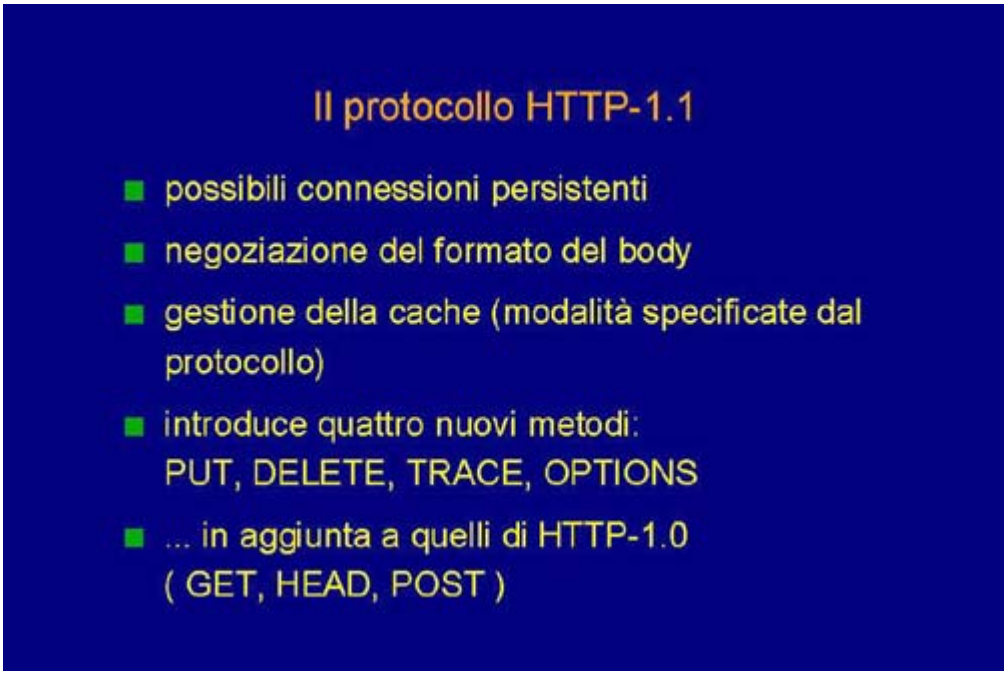
Il protocollo HTTP-1.0



Sia che si tratti di HTML, sia di XML, in entrambi i casi questi dati verranno trasportati tramite il protocollo HTTP; tale protocollo nella versione 1.0 (HTTP-1.0) funziona nel seguente modo: quando il browser desidera prelevare dei dati dal server istituisce una connessione, tipicamente questo è un canale TCP relativo alla porta 80 del server. Una volta che il collegamento è stato stabilito, il browser manda su questo canale la sua richiesta di informazione, tipicamente questo avviene sottoforma di un comando GET, cioè si vuol prendere dei dati. Il server riceve questa richiesta di informazioni e fornisce la risposta, anche qui tipicamente tale risposta è sottoforma di file HTML , oppure se siamo più moderni XML. Non appena la risposta è arrivata, il server chiude il collegamento. Notate che ipoteticamente sarebbe possibile che il browser avesse altre domande da fare nei confronti del server, ma poiché i server Web sono tipicamente molto sovraccarichi è stata scelta questa strada: il browser può fare soltanto una domanda alla volta per evitare di

sovraccaricare il server tenendo aperti dei canali che non si sa quando verranno utilizzati. Quindi se un browser accede ripetutamente ad un server Web, con il protocollo HTTP-1.0 è obbligato ogni volta ad aprire e chiudere i canali effettuando su ogni canale una sola richiesta per volta. Il protocollo HTTP-1.0 è oggi evoluto verso la versione HTTP-1.1.

Il protocollo HTTP-1.1



Il protocollo HTTP-1.1 protocollo permette, sotto il controllo del sistemista che gestisce il server, di creare connessioni persistenti, ossia permette al browser di creare un canale e di mantenerlo aperto facendo più richieste sul medesimo canale; questo è vantaggioso perché allevia il lavoro da parte del client e stressa di meno la rete, ma è pericoloso perché potenzialmente porta un grosso sovraccarico da parte del server. Nella versione 1.1 è anche possibile, per browser e server, negoziare il formato dei dati, ossia del body delle informazioni che vengono scambiate, in questo modo se il server ha a disposizione i dati in più formati diversi, è possibile fornire al client i dati nel formato che lui è meglio in grado di accettare. Un'altra innovazione è stata la gestione della cache; voi sapete che la cache è uno dei meccanismi con cui i browser o i nodi intermedi, cosiddetti proxy, possono tener copia locale dei dati, per evitare ogni volta di andarli a prendere dal server. In questo caso è possibile specificare all'interno del protocollo, quindi da parte di chi gestisce il server Web, qual è la modalità per la gestione della cache. Inoltre sono stati introdotti quattro nuovi metodi, cioè quattro nuove operazioni che il browser può fare nei confronti del server; originariamente le operazioni erano soltanto quelle di prendere un documento, prendere solo l'intestazione di un documento, oppure mandare delle informazioni semplici, tipicamente il risultato di un input, al server. Nell'HTTP 1.1 sono state aggiunte a queste anche le operazioni chiamate PUT, che permettono di mettere un intero documento sul server, DELETE, che permettono di cancellarlo, TRACE, che serve a vedere qual è la sequenza dei proxy tra il browser e il server e OPTIONS, che servono a conoscere quali sono le opzioni del canale HTTP presente. In questo modo noi stiamo evolvendo verso un'architettura molto più ricca di possibilità. Quindi l'unione del directory, con il server di tipo Web, con il linguaggio che da HTML sta evolvendo verso XML, sono oggigiorno i tre cardini portanti della realizzazione di qualunque sistema informativo distribuito di tipo aperto.