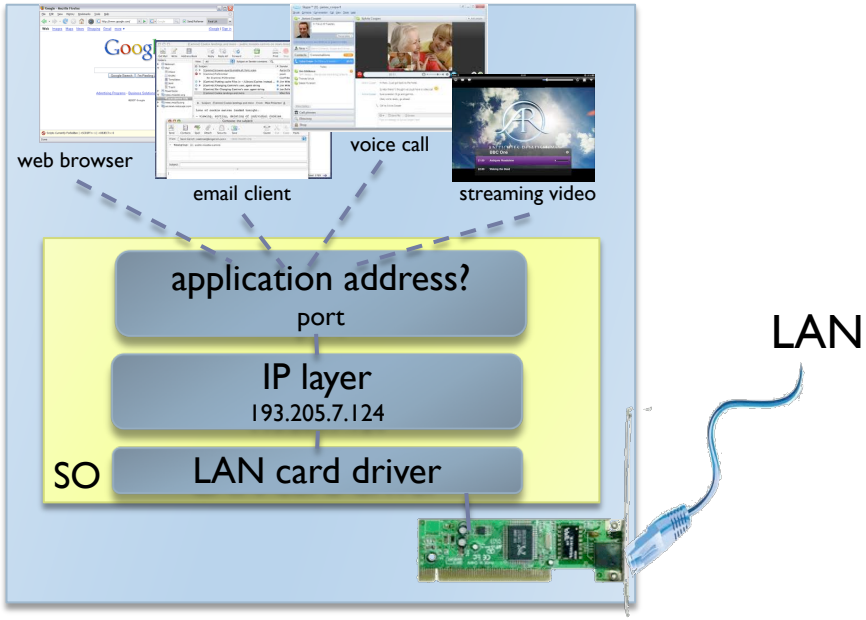


Reti di Calcolatori

Application Layer

Application Layer

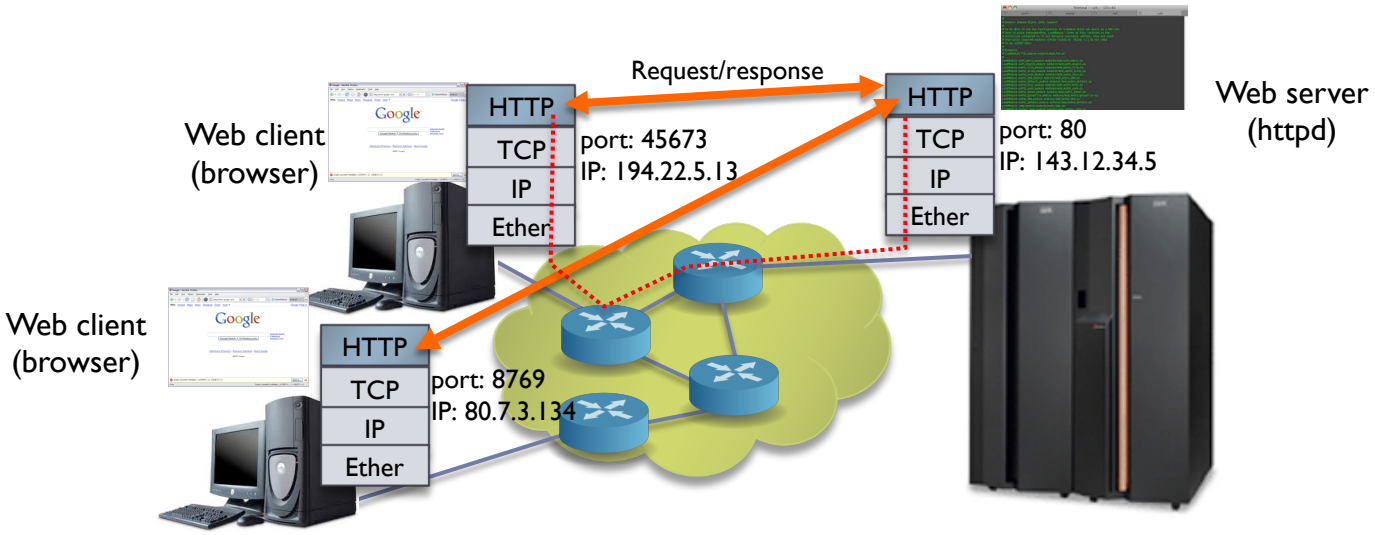
- Programmi applicativi che utilizzano i servizi di rete
  - Sono spesso basati sul modello client/server
  - I protocolli applicativi permettono di standardizzare la comunicazione in modo da rendere interoperabili programmi sviluppati indipendentemente
    - specificano il formato dei messaggi di richiesta e risposta che sono scambiati fra gli agenti software coinvolti
    - definiscono il meccanismo di trasporto utilizzato (TCP o UDP)
  - Il servizio di trasporto utilizzato dipende dal tipo di applicazione
    - UDP è adatto per le applicazioni che richiedono bassa latenza (ritardi) e overhead (dati in più da trasmettere) ma possono tollerare la perdita di informazioni (es. trasmissione video o audio)
    - TCP si usa quando è necessario garantire che tutti i dati trasmessi arrivino tollerando eventuali ritardi (es. trasferimento pagine Web)
  - nessuno dei due protocolli fornisce garanzie di temporizzazione

Applicazioni di rete

- Esistono numerose applicazioni e protocolli applicativi
  - DNS (Domain Name System)
    Risoluzione dei nomi degli host
  - SMTP (Simple Mail Transfer Protocol) e email
    sendmail o postfix
  - POP3 (Post Office Protocol) e IMAP (Interactive Mail Access Protocol)
    Accesso remoto alle caselle di posta elettronica
  - FTP (File Transfer Protocol)
    Trasferimento file remoti
  - SNMP (Simple Network Management Protocol)
    Gestione di apparati di rete
  - HTTP (HyperText Transfer Protocol) e il WWW
    Server web e browsers
  - telnet, rlogin, rcp, ssh, sftp, nfs, lpd,bittorrent, applicazioni P2P (Peer to Peer), applicazioni per streaming video/audio, ...

Architetture delle applicazioni di rete 1

- Client-Server
  - Il server è un applicativo sempre attivo in esecuzione su un host
    - L'indirizzo IP (nome DNS) dell'host server deve essere noto
    - Il server è in ascolto su una porta (TCP o UDP) nota (standard)
  - I client sono applicativi in esecuzione su altri host della rete
    - I client sono saltuariamente attivi
    - Non è richiesto di conoscere a priori l'indirizzo IP e la porta usata dal client

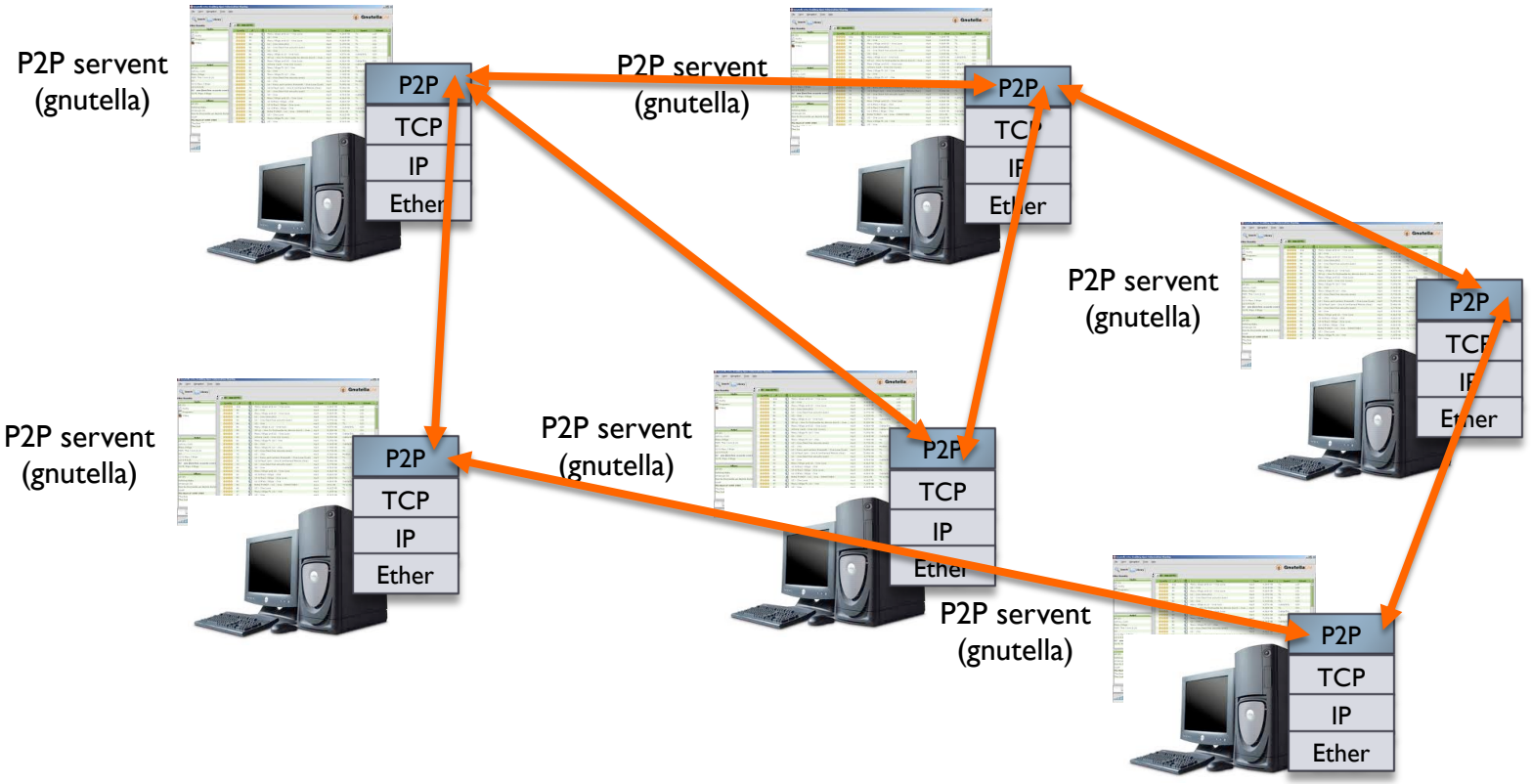


Architetture delle applicazioni di rete 2

- Peer to Peer (P2P)
  - Tutti i nodi (peers) hanno in linea di principio la stessa importanza
  - Ogni nodo è client e server (servent)
  - Ogni peer condivide delle risorse (dati/potenza di calcolo)
  - Possono comunque essere presenti server centralizzati o nodi con funzionalità diverse rispetto agli altri (supernodi)
  - Il numero di nodi può essere dell'ordine delle centinaia di migliaia
  - I nodi sono dinamici e autonomi
    - un nodo può entrare o uscire dalla rete in ogni momento
  - gli indirizzi dei nodi non sono noti a priori e quindi devono essere definiti dei meccanismi di discovery
    - su una LAN si possono usare richieste broadcast (UDP)
    - su Internet occorre un server che indicizzi i peer (o parte di essi)

Rete P2P

- I peer sono connessi da una rete a livello applicativo
  - Un peer mantiene più connessioni TCP con altri peer
  - La rete di connessioni è usata per la gestione della rete (es. propagare le ricerche)

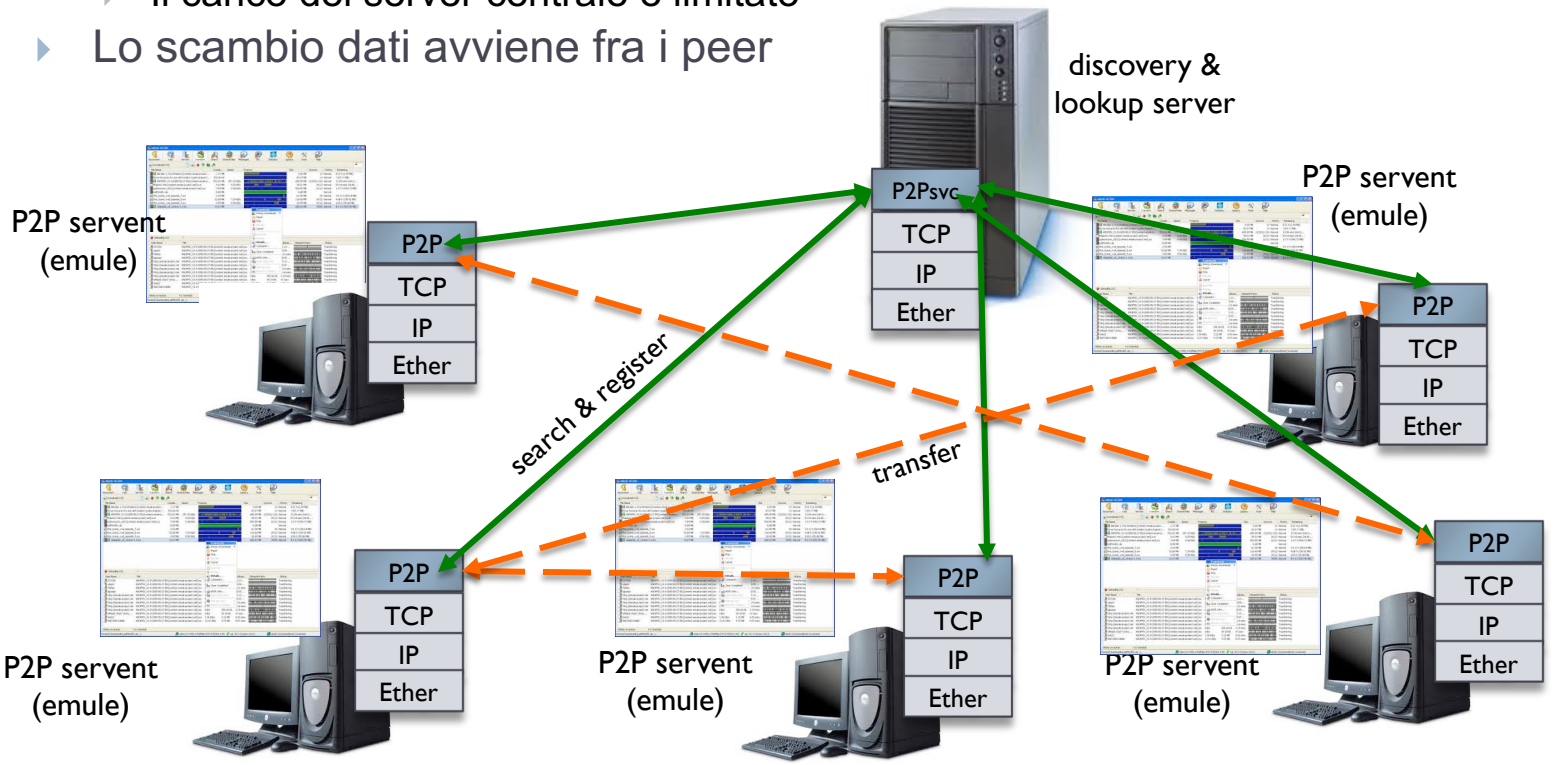


Reti P2P: Pro & Cons

- Il vantaggio delle reti P2P pure (senza server centralizzati) è la scabilità
  - tutti i peer si dividono il carico
  - ogni peer aumenta la domanda ma anche la capacità di servizio
- La gestione delle reti P2P pure è complessa
  - gestione delle ricerche in modo distribuito (mancanza di un indice centrale): *chi ha la risorsa che serve*?
    - La ricerca con flooding (propagazione a tutti i peer) crea un traffico elevato
  - gestione della rete in modo robusto: *come si trovano i peer a cui connettersi*? *Come si garantisce l'integrità della rete rispetto alla scomparsa di peer*?
- In genere le reti P2P pure non sono efficienti
  - es. Gnutella

Architetture ibride

- E' possibile combinare l'architettura P2P con quella client server (es. bittorrent, e-mule, skype)
  - Un server centrale fornisce alcuni servizi (indice per le ricerche, discovery dei peer)
    - Il carico del server centrale è limitato
  - Lo scambio dati avviene fra i peer



Domain Name System (DNS)

- E' un servizio che permette di convertire i nomi (DNS) degli host su Internet nei corrispondenti indirizzi IP
  - Il nome DNS (è una stringa di caratteri) più facile da ricordare (e che dà più informazioni) rispetto ad un numero a 32 bit

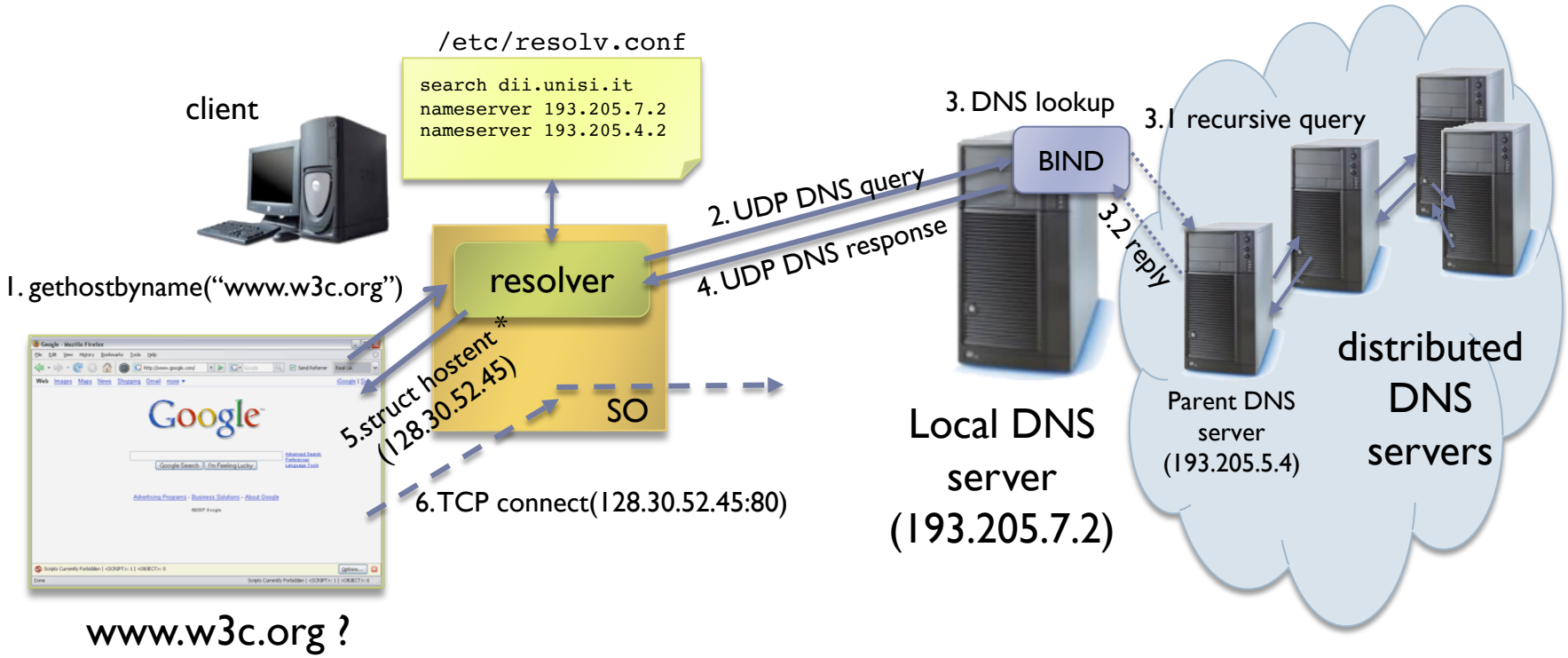
  - Senza il DNS la rete Internet funzionerebbe ugualmente ma non sarebbe usabile facilmente dagli utenti
  - Il DNS è un servizio critico per il funzionamento di Internet
  - Per scambiare dati due host usano gli indirizzi IP: prima di poter contattare un host l'indirizzo DNS deve essere trasformato nel suo IP corrispondente (DNS lookup)

Struttura del DNS

- Il DNS è un'applicazione client/server
  - è un database distribuito gestito da un insieme di server DNS organizzati in modo gerarchico
  - definisce un protocollo applicativo con cui è possibile interrogare il database per mezzo di pacchetti UDP (porta 53)
  - Il client (resolver) è un servizio del Sistema Operativo
    - Occorre configurare il resolver specificando l'indirizzo IP del server DNS (Name server) del dominio locale (anche più di uno)
    - In UNIX è specificato nel file di sistema `/etc/resolv.conf`
    - Esiste una API C del resolver: `gethostbyname()`, `gethostbyaddr()`
    - Quando un'applicazione deve convertire un nome DNS in indirizzo IP, utilizza il resolver che con il protocollo DNS contatta il server DNS
    - `nslookup` è un programma client per interrogare il DNS

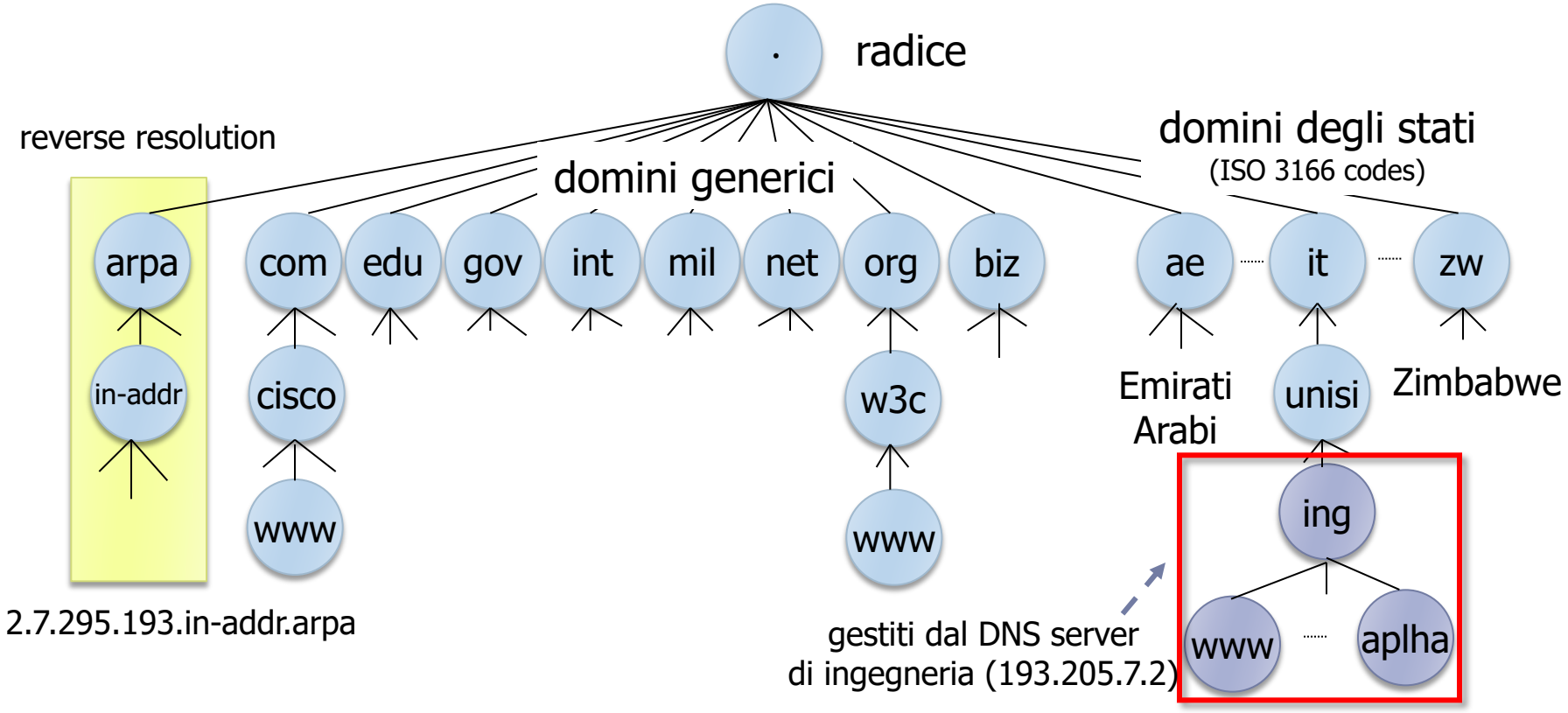
Risoluzione dei nomi DNS

- Il DNS è usato da altri protocolli applicativi quali ad esempio HTTP, SMTP
  - La richiesta DNS introduce di fatto un ritardo



Domain Name Space

- Lo spazio dei nomi è organizzato in modo gerarchico
  - è un abero di nomi di dominio (in parte definito da standard)
  - ciascun nodo o foglia ha associati zero o più resource records



I nomi di dominio

- Un nome di dominio è una stringa composta da una o più parti (label)
  - le label sono concatenate col carattere .
  - ciascuna label è lunga al massimo 63 caratteri e il nome completo non può superare i 255 caratteri (byte)
    - I caratteri amessi sono un sottoinsieme della codifica ASCII (LDH – Letters Digit Hyphen ma non possono iniziare col carattere -)
    - I nomi DNS sono case-insensitive
  - Un nome di dominio assoluto (fully qualified domain name - FQDN) termina con un punto (es. **`www.sun.com.`**)
    - I nomi che non terminano con un punto sono relativi (es. **`alpha`** è relativo al dominio in cui si trova l'host ovvero **`ing.unisi.it.`**)
  - la label più a destra identifica il top-level domain (es. **`.com`**)
  - un hostname è un nome DNS a cui è associato almeno un IP

Host & mail server aliasing

- **Host aliasing**
  - Un host può avere uno a più sinonimi (alias)
  - Il nome DNS principale è detto canonico

```
www.w3c.org canonical name = dolph.w3.org.
Name: dolph.w3.org
Address: 128.30.52.45
```

- **Mail server aliasing**
  - Il DNS permette una risoluzione dei nomi specifica per i domini di posta elettronica (record di tipo **MX** – **Mail eXchange**)
  - Il nome del dominio può essere associato al server di posta elettronica per rendere più leggibili gli indirizzi email

```
➢set type=MX
➢unisi.it
unisi.it        mail exchanger = 10 antispam.unisi.it.
unisi.it        mail exchanger = 10 antispam2.unisi.it.
```

Associazioni "uno-a-molti"

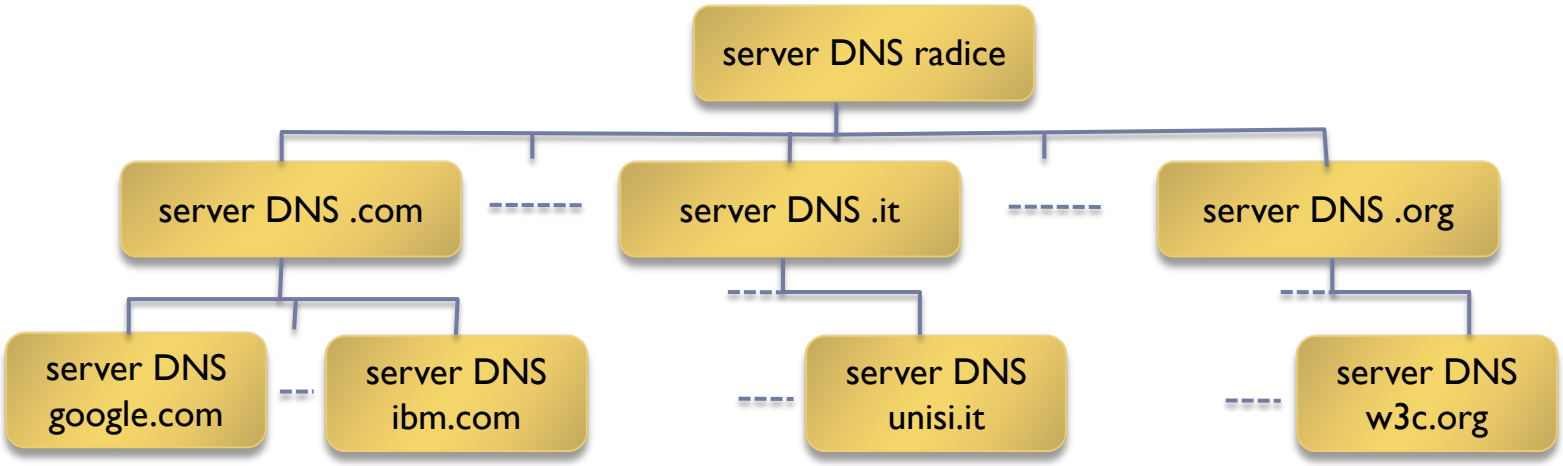
- Il DNS può essere utilizzato per distribuire il carico fra più server con IP diversi
  - Ad un hostname canonico vengono associati più indirizzi IP
  - Il server DNS risolve il nome fornendo l'insieme di indirizzi ma ruota l'ordinamento ad ogni risposta
    - In genere il resolver sceglie il primo indirizzo della lista per cui le richieste sono distribuite a rotazione fra i vari host

```
www.google.com   canonical name = www.l.google.com.
Name:        www.l.google.com
Address: 173.194.35.146
Name:        www.l.google.com
Address: 173.194.35.147
Name:        www.l.google.com
Address: 173.194.35.148
Name:        www.l.google.com
Address: 173.194.35.144
Name:        www.l.google.com
Address: 173.194.35.145
```

Il DNS come database distribuito

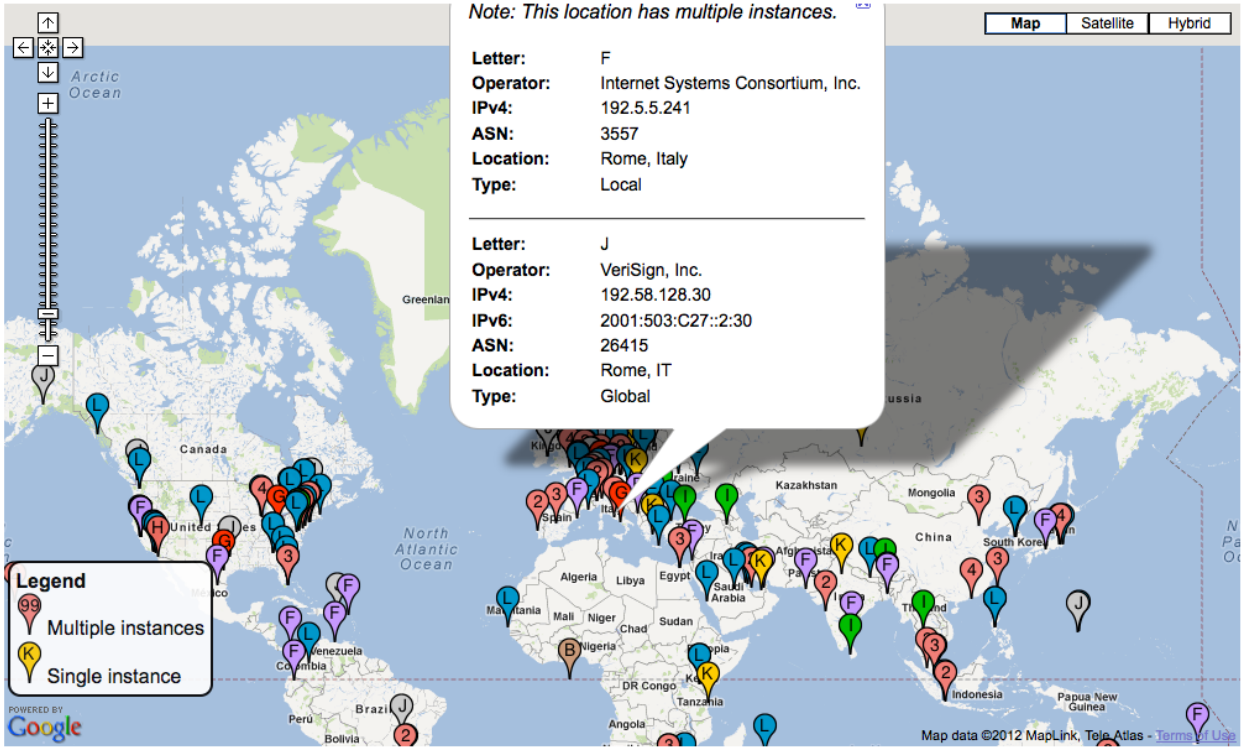
- Il servizio DNS è realizzato con un gran numero di server DNS distribuiti sulla rete
  - Un approccio centralizzato per il DNS non funzionerebbe
    - Il server centrale sarebbe un collo di bottiglia sia per eventuali guasti che per la gestione del traffico richiesto (ogni client su rete genera numerose richieste DNS)
    - L'accesso al server centrale da punti diversi della rete potrebbe richiedere latenze anche lunghe
    - Tenere aggiornato il database sarebbe complesso perché il numero di record è molto elevato. Inoltre i record sono amministrati da numerosi soggetti eterogenei (aziende, organizzazioni, università,..)
  - I server DNS sono organizzati in modo gerarchico e sono distribuiti nel mondo
    - ciascun server DNS ha competenza solo per un sottoinsieme di hostname

Struttura gerarchica dei DNS server



- Server radice (root servers)
  - Esistono 13 server radice etichettati da A a M
- Server top-level domain (TLD)
  - 20 domini generici e 248 livelli radici per le nazioni (dato 2009)
- Server di competenza (authoritative server)
  - server che gestisce effettivamente i record per una zona
  - I nomi di dominio devono essere registrati presso una registration authority fornendo i riferimenti ai name server di competenza

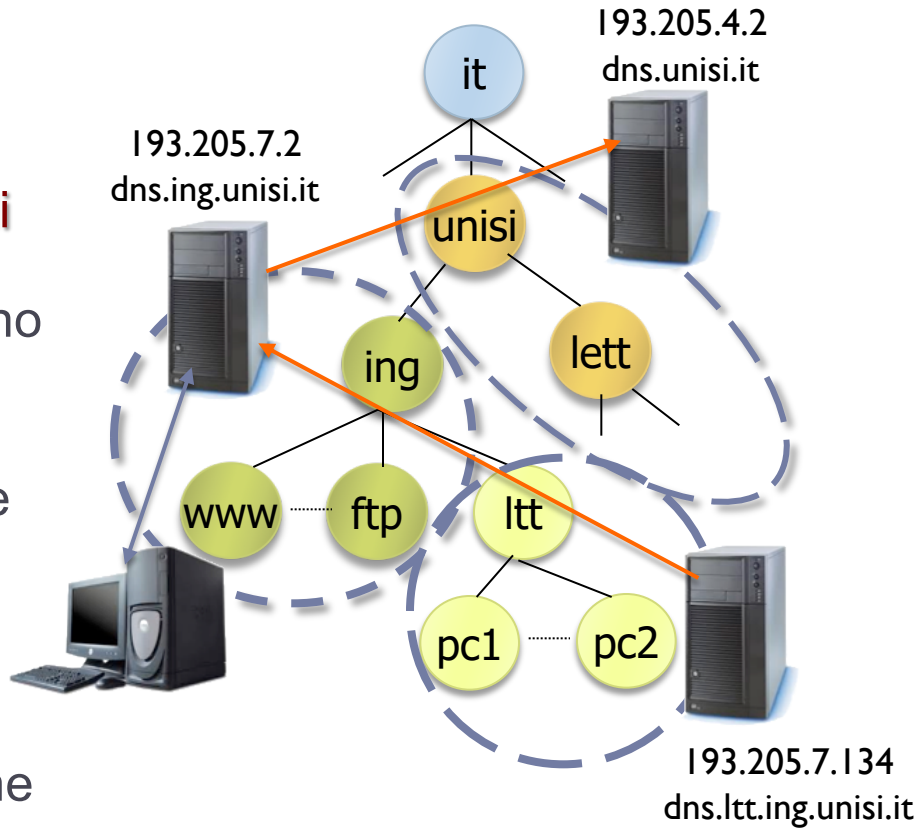
Root servers (www.root-servers.org)



- I 13 root server (A. root-servers.net – M.root-servers.net) hanno repliche anche in zone geografiche diverse
- Il root zone file contiene la lista dei nomi e indirizzi IP per i server di competenza dei top level domains (circa 200kb)
- Il root server fornisce il riferimento al server TLD competente per un dato dominio

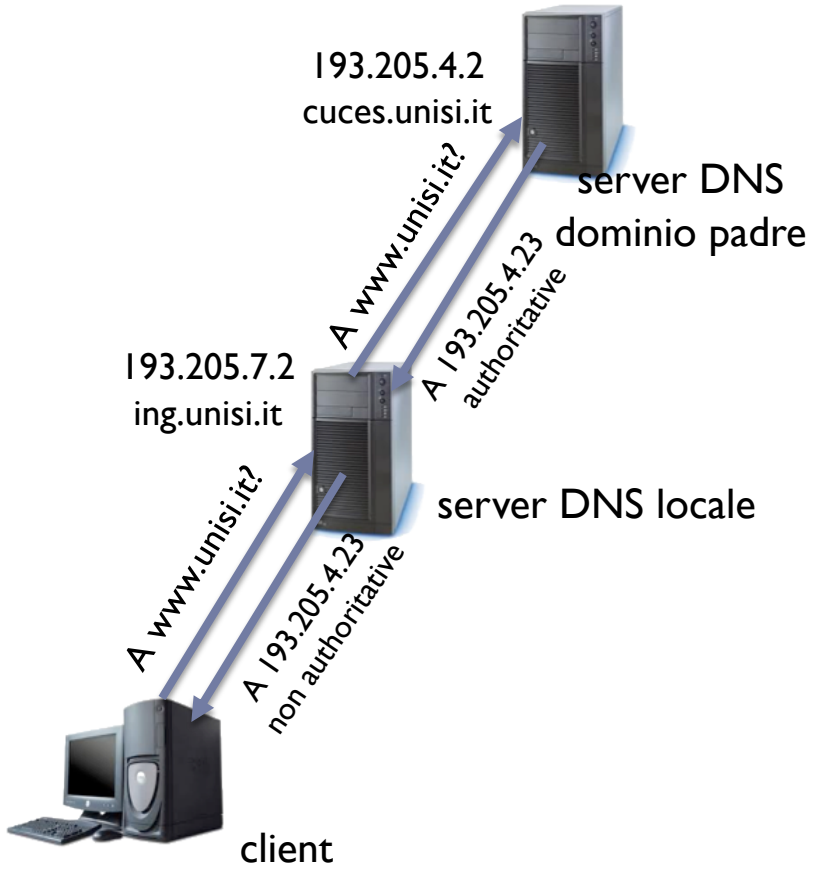
Zone & Authoritative Name Servers

- Lo spazio dei nomi di dominio è diviso in zone
  - Una zona corrisponde ad un sottoalbero della gerarchia dei nomi
  - Ogni zona ha un server DNS di competenza primario ed eventuali server secondari (sono repliche sincronizzate col primario)
  - Un authoritative server fornisce risposte che derivano da una sorgente diretta (es. amministratore di rete)
  - Un server DNS può essere authoritative server per più zone



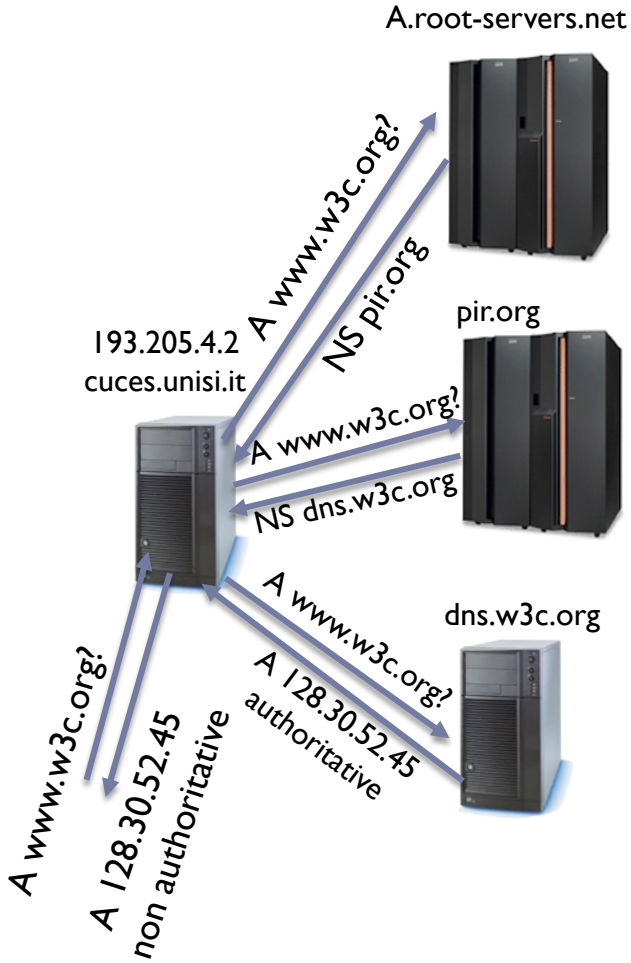
Interrogazione ricorsiva

- Il resolver chiede la risoluzione di un nome al server DNS locale con **modalità ricorsiva**
  - Se il nome è locale si ottiene un authoritative record
  - Se il nome appartiene ad una zona non gestita direttamente dal server locale si innesca un'interrogazione ricorsiva (o iterativa) ad altri nameserver
  - Il risultato è il RR richiesto o un errore
  - La modalità ricorsiva viene definita da un bit di flag nella richiesta



Interrogazione iterativa

- In un'interrogazione iterativa il client richiede al server la migliore risposta che può dare
  - Se il nome richiesto non è noto il server fornisce un riferimento al name server di competenza per un livello più basso dello spazio dei nomi
    - Al primo passo si contatta un root server che fornisce il riferimento al server TLD di competenza (es. .org)
    - Si contatta il server TLD che fornisce il riferimento al name server del dominio specifico (es. w3c.org)
    - Il name server invia la query al name server di dominio che o risponde con un RR con l'IP o fornisce un ulteriore riferimento ad un name server di sottodominio



Esempi con nslookup

```
> www.ing.unisi.it
Server:     193.205.7.2
Address:   193.205.7.2#53

www.ing.unisi.itcanonical name = firewall.ing.unisi.it.
Name:      firewall.ing.unisi.it
Address: 193.205.7.2
```

Authoritative answer

```
> www.unisi.it
Server:     193.205.7.2
Address:   193.205.7.2#53

Non-authoritative answer:
www.unisi.it      canonical name = pico.unisi.it.
Name:      pico.unisi.it
Address: 193.205.4.23
```

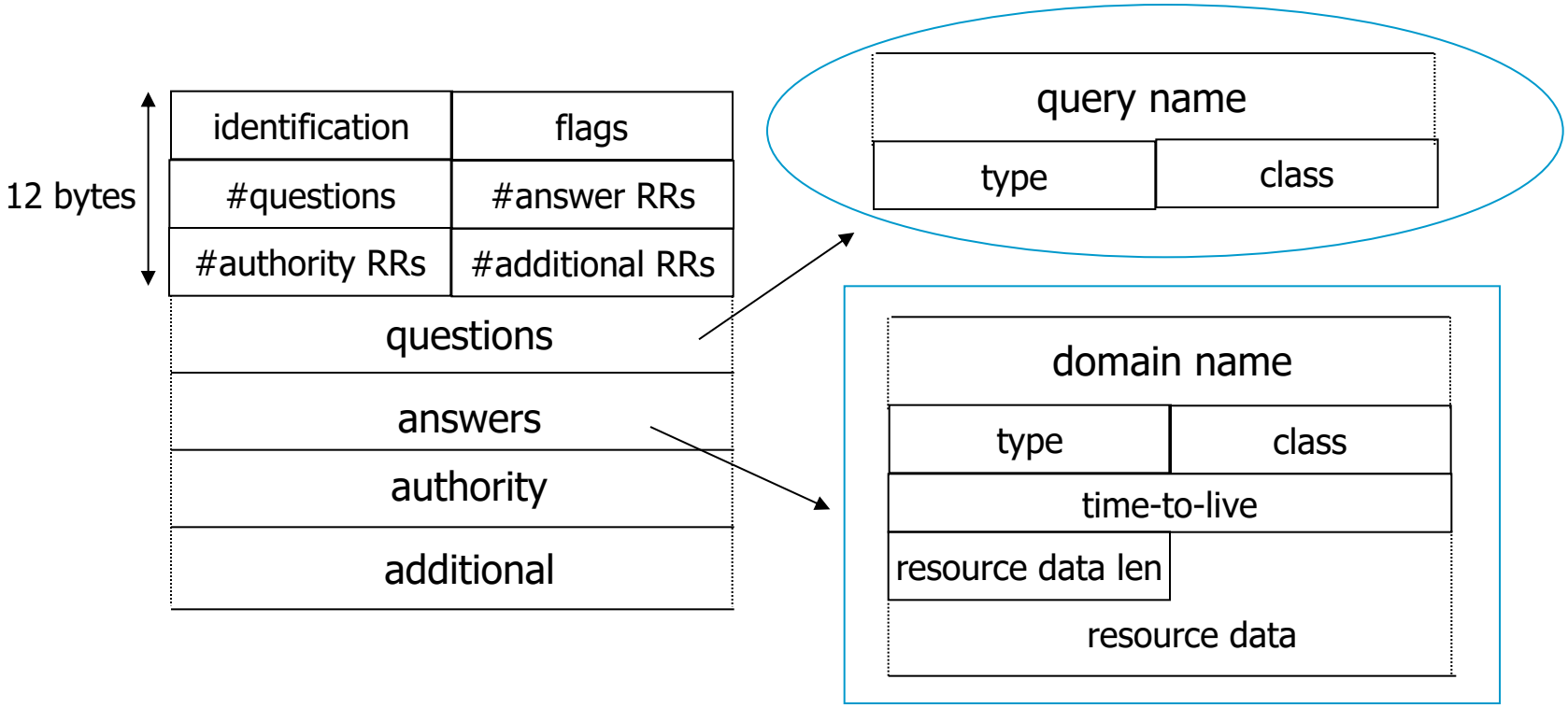
Non Authoritative answer ottenuta con interrogazione ricorsiva

Cache DNS

- Il caching DNS permette di ridurre
  - la latenza nelle risposte
  - il numero di messaggi necessari ad ottenere la risposta
- I server DNS e il resolver memorizzano in una cache locale le risposte ricevute
  - Se una nuova richiesta è relativa un RR presente in cache la risposta può essere inoltrata senza ulteriori interrogazioni (HIT)
  - I RR nella cache vengono comunque invalidati dopo che è scaduto il loro Time To Live (TTL) per gestire il fatto che le associazioni IP-hostname possono essere dinamiche
  - I RR in cache comprendono anche gli indirizzi NS dei server TLD o di dominio in modo che le interrogazioni iterative non hanno necessità di contattare i relativi server DNS

Formato dei messaggi DNS

- Il messaggio di richiesta/risposta è inviato/ricevuto come pacchetto UDP
- Il messaggio può contenere una o più richieste o risposte sotto forma di Resource Record (RR)



Tipi di Resource Records (RRs)

- **A**
  - *Nome*: hostname - *Valore:* Indirizzo IP
- **NS**
  - *Nome:* dominio – *Valore:* hostname del server di competenza
- **PTR**
  - *Nome:* indirizzo IP – *Valore:* hostname
- **CNAME**
  - *Nome:* alias – *Valore:* nome canonico
- **HINFO**
  - *Nome:* hostname – *Valore:* Informazioni sull'host (Tipo, Sistema Operativo, ...)
- **MX**
  - *Nome:* hostname email – *Valore:* nome canonico del server email

DNS database file

```
@   IN      SOA     alpha.dii.unisi.it.
    postmaster.alpha.dii.unisi.it. (
                    2000072502; serial
                    86400 ; refresh
                    1800 ; retry
                    2592000 ; expire
                    4320000 ; default ttl )

@   IN      NS      alpha.dii.unisi.it.
@   IN      NS      cuces.unisi.it.

alpha           IN  A       193.205.7.2
                IN  HINFO   Pentium II Linux
mailsrv         IN  CNAME   alpha
www             IN  CNAME   alpha
ing.unisi.it    IN  MX      0 alpha
```

Il servizio di posta elettronica @

- L'email rappresenta un sistema di comunicazione asincrona
  - L'invio da parte dell'utente mittente e la ricezione da parte dell'utente destinatario dei messaggio non sono sincronizzati
  - Il sistema di gestione dei messaggi deve prevedere un meccanismo di memorizzazione dei messaggi inviati ma ancora non letti
- L'infrastruttura del servizio email prevede
  - La definizione di un sistema di indirizzamento per le "caselle postali" (mailbox) a cui sono inviati i messaggi
  - Un sistema di agenti software per inviare, ricevere, e consegnare i messaggi (agenti utente e agenti di trasferimento) con il relativo protocollo di comunicazione (Simple Mail Transfer Protocol SMTP)
  - Uno standard per il formato dei messaggi (busta)

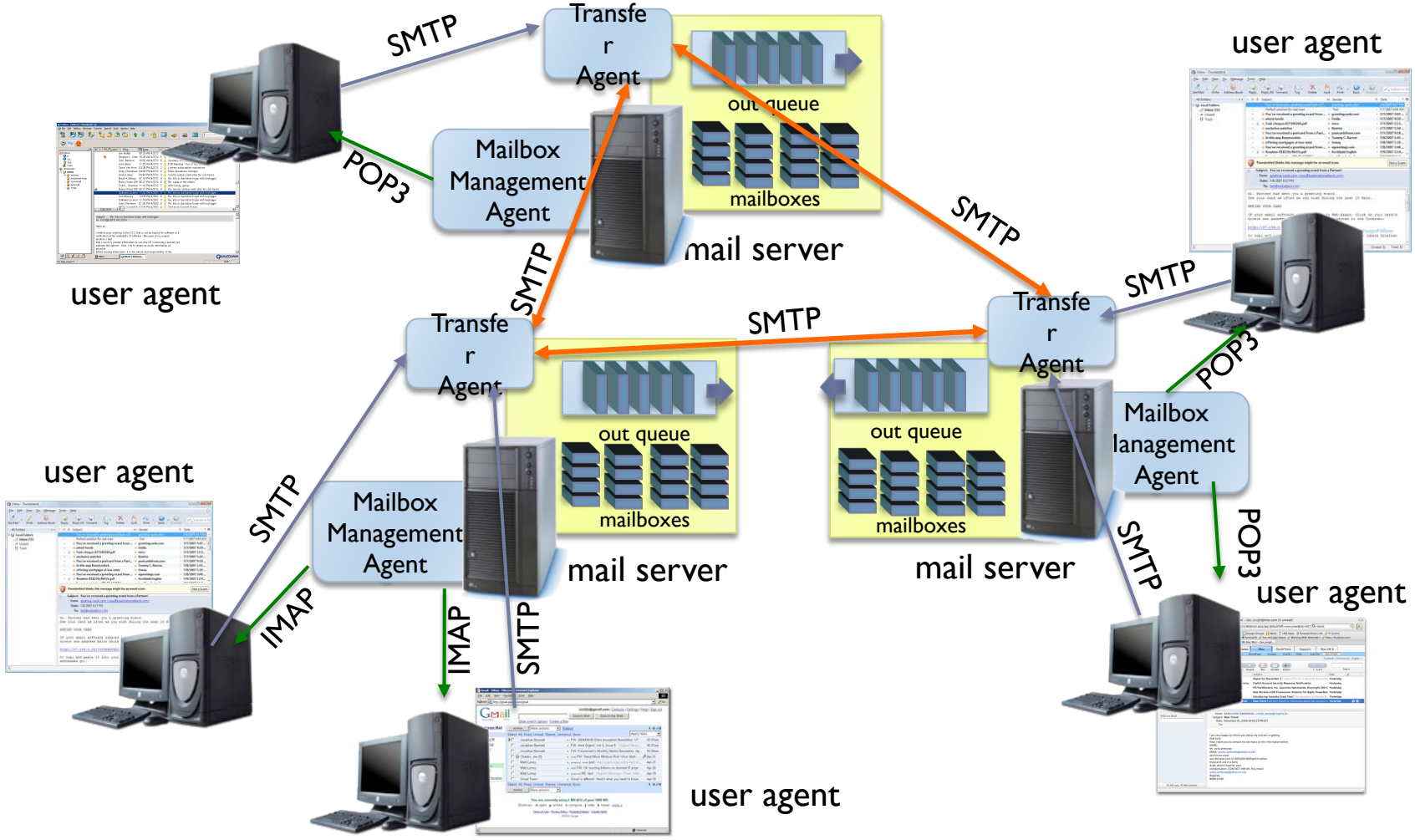
Indirizzi email @

- Un indirizzo email individua una mailbox su un host nella rete Internet
  - La mailbox è un file di testo in una directory specifica nel server (es. in Unix è normalmente `/var/spool/mail/`*`utente`*)
  - I messaggi di posta in arrivo sono accodati (append) al file di mailbox
  - L'utente può accedere alla posta localmente leggendo il file mailbox o attraverso protocolli per l'accesso remoto alla mailbox (POP3 o IMAP)
  - Un indirizzo email ha la struttura

`utente@host.dominio`

  - **`utente`** è l'identificativo della mailbox (si possono avere alias)
  - **`host.dominio`** è risolto dal DNS con una query di tipo MX

Infrastruttura @ email



Agenti @ email

- **Mail User Agent (MUA)**
  - Programmi per leggere, comporre i messaggi e gestire le mailbox
    - eudora, thunderbird, outlook, pine, mac mail, web mail,..
  - Utilizzano un server SMTP per la spedizione dei messaggi (eventualmente con autenticazione)
  - Gestiscono di mailbox remote interagendo con il server usando il protocollo POP3 o IMAP

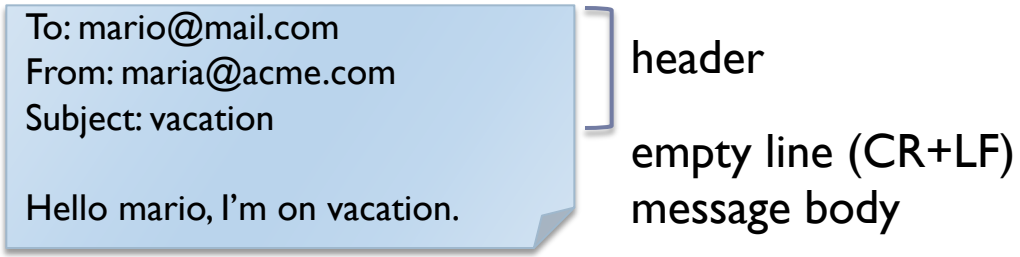
- **Message Transfer Agent (MTA)**
  - Gestiscono il trasferimento dei messaggi dal mittente alla destinazione e la ricezione dei messaggi sui server di posta
    - Inviare un messaggio corrisponde a trasferire un file di testo
  - Sono servizi di sistema che usano il protocollo SMTP su una connessione TCP (porta server 25) per trasferire i messaggi
    - sendmail, postfix, ...

Consegna di un messaggio

- Generalmente la consegna di un messaggio è diretta
  - L'user agent trasferisce il messaggio sul suo MTA (SMTP server)
  - Il MTA eventualmente accoda il messaggio nella coda di uscita
  - Il MTA mittente estrae l'hostname dall'indirizzo email del destinatario e apre una connessione TCP sulla porta 25 di tale host contattando Il MTA destinatario
  - Utilizzando il protocollo SMTP il messaggio è trasferito fra i due MTA
  - l'MTA destinatario accoda il messaggio nella mailbox dell'utente specificato
- Gli MTA possono anche funzionare da Relay Agent
  - Il messaggio può transitare da MTA intermedi che accettano l'inoltro di messaggi verso altri MTA
  - In genere la funzionalità Relay è disabilitata perché può essere usata per inviare SPAM

Formato dei messaggi @ email

- Il formato dei messaggi è definito in RFC 821 - RFC 822
  - Un messaggio è un testo con codifica dei caratteri ASCII a 7 bit, ovvero utilizza solo i codici da 0 a 127
  - Il messaggio è strutturato in due parti separate da una linea vuota



  - L'intestazione contiene meta-informazioni che descrivono proprietà del messaggio
    - è un insieme di linee di testo con codifica ASCII con la struttura

`campo`: `valore`

    - Le intestazioni possibili e la loro semantica sono definite e standardizzate in RFC 822, 2045 e 2046 (estensioni MIME)

Intestazione dei messaggi @ email

- Le intestazioni sono inserite dall'user agent e/o dai MTA
  - Alcune intestazioni sono informative, alcune servono per la consegna, altre per l'interpretazione del corpo del messaggio
  - Alcuni esempi
    - **`To:`** indirizzo DNS del destinatario (o destinatari)
    - **`Cc:`** copia in carta carbone (indirizzi "per conoscenza")
    - **`Bcc:`** copia in carta carbone con indirizzi invisibili ai riceventi
    - **`From:`** indirizzo di ha scritto il messaggio (necessario)
    - **`Sender:`** indirizzo di chi ha inviato il messaggio (opzionale)
    - **`Subject:`** oggetto del messaggio
    - **`Date:`** data e ora in cui il messaggio è stato inviato
    - **`Reply-to:`** indirizzo al quale inviare le risposte (reply) se diverso da From:
    - **`Received:`** indica l'agente che ha ricevuto il messaggio insieme a un timestamp. Viene inserito da ogni MTA durante il trasferimento
    - **`Message-Id:`** numero unico che individua il messaggio sul MTA origine

Un esempio di intestazione

```
Return-Path: <BWerner@computer.org>
Delivered-To: maggini@dii.unisi.it
Received: from firewall.dii.unisi.it (firewall.dii.unisi.it [10.0.0.1])
    by alpha.dii.unisi.it (Postfix) with ESMTP id B763218337
    for <maggini@dii.unisi.it>; Sat, 19 May 2001 03:02:57 +0200 (CEST)
Received: from sendmailout.computer.org (unknown [206.99.234.2])
    by firewall.dii.unisi.it (Postfix) with ESMTP id 6A9053C0BC
    for <maggini@dii.unisi.it>; Sat, 19 May 2001 02:48:10 +0200 (CEST)
Received: from cray.computer.org ([63.84.223.121])
    by sendmailout.computer.org (Build 101 8.9.3/NT-8.9.3) with ESMTP
    id UAA01113; Fri, 18 May 2001 20:03:57 -0400
Subject: ICDAR'01: Your Author Kit
To: ld47@csgz.edu.cn, maggini@ing.unisi.it
Cc: Karl.Tombre@loria.fr, haralick@gc.cuny.edu
X-Mailer: Lotus Notes Release 5.0.4  June 8, 2000
Message-ID: <OF1A96792E.5B9B2F1A-ON88256A50.00814155@computer.org>
From: BWerner@computer.org
Date: Fri, 18 May 2001 17:03:57 -0700
X-MIMETrack: Serialize by Router on Cray/IEEE Computer Society(Release
    5.0.6a |January 17, 2001) at 05/18/2001 05:11:59 PM
MIME-Version: 1.0
Content-type: multipart/mixed;
    Boundary="0__=88256A50008141558f9e8a93df938690918c88256A5000814155"
Content-Disposition: inline
```

Codifica dei caratteri su ASCII 7 bit 1

- Il messaggio email può contenere solo caratteri con codifica NVT ASCII a 7 bit
  - sono trasmessi su 8 bit con il bit più significativo a 0 (storicamente era utilizzato per il bit di controllo parità)
  - per trasmettere caratteri non NVT ASCII occorre utilizzare una codifica opportuna
    - trasmissione di dati binari (attachment) come ad esempio immagini, documenti, file compressi
    - trasmissione di testi che contengono caratteri non ASCII base (ASCII esteso per il set di caratteri con codici fra 128 e 255) o set di caratteri diversi (Unicode, Japanese, Chinese, ...)
  - Nell'intestazione del messaggio si deve poter indicare
    - Il tipo di dati contenuto e il metodo di codifica su ASCII 7 bit usato
    - Nel caso di testi, il set di caratteri utilizzato (Character Encoding - charset)

Character Encoding & ASCII esteso

- Un Character Encoding è l'assegnazione di un codice binario ad ogni carattere di un alfabeto
  - il charset è un'informazione fondamentale da rendere nota nello scambio dati per questo è sempre prevista una modalità per associare un meta-dato che definisce il charset utilizzato
    - se i dati non sono associati al charset appropriato non si è in grado di visualizzarli correttamente
  - ISO 8859 descrive estensioni al set ASCII a 7 bit per linguaggi che utilizzano caratteri non presenti nella codifica base per l'Inglese
    - permette ulteriori 96 caratteri stampabili oltre a quelli ASCII standard 7 bit
    - ne esistono versioni diverse che codificano set di caratteri diversi, es.
      - ISO-8859-1 Latin-1 (copertura completa dell'Italiano)
      - ISO-8859-10 Latin-6 (linguaggi nordici)

ASCII e ASCII esteso

ASCII (1977/1986)

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_ 0	NUL 0000	SOH 0001	STX 0002	ETX 0003	EOT 0004	ENQ 0005	ACK 0006	BEL 0007	BS 0008	HT 0009	LF 000A	VT 000B	FF 000C	CR 000D	SO 000E	SI 000F
1_ 16	DLE 0010	DC1 0011	DC2 0012	DC3 0013	DC4 0014	NAK 0015	SYN 0016	ETB 0017	CAN 0018	EM 0019	SUB 001A	ESC 001B	FS 001C	GS 001D	RS 001E	US 001F
2_ 32	SP 0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3_ 48	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
4_ 64	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
5_ 80	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C	] 005D	^ 005E	_ 005F
6_ 96	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
7_ 112	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	| 007C	} 007D	~ 007E	DEL 007F

ISO/IEC 8859-1

Hex	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_																
1_																
2_	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3_	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4_	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5_	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6_	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7_	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8_																
9_																
A_		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
B_	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C_	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D_	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E_	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F_	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

- Set esteso di caratteri
  - Codici > 127
  - Es. è E8 (1110 1000)
  - E 8

Codifica dei caratteri su ASCII 7 bit 2

- Una stringa nell'intestazione può essere codificata come

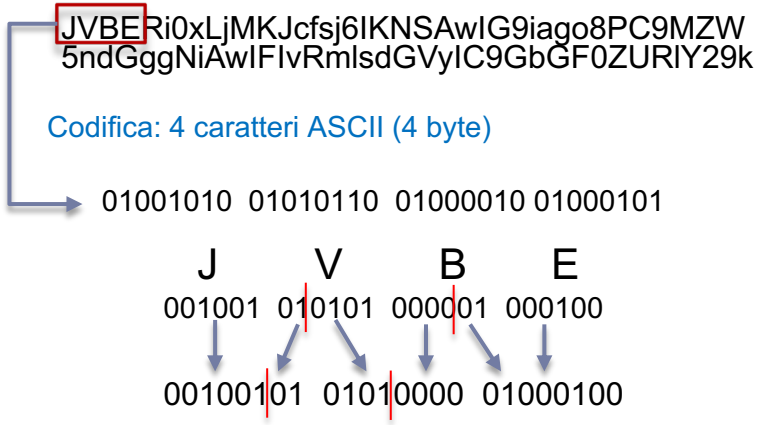
```
=?charset?encoding?encoded-text?=
```

  - charset è una stringa standard che descrive il character encoding
  - encoding è il tipo di codifica usata per mappare i caratteri non ASCII a 7 bit ovvero quelli con valore nel byte > 127
    - La codifica quoted-printable (**Q**) codifica i caratteri con codice > 127 con **=** seguito dai due caratteri delle cifre esadecimali del codice del carattere (`0-9`, `A-F`). Ad esempio è si codifica con `=E8`.
    - La codifica base64 (**B**) codifica 3 caratteri consecutivi (24 bit) come 4 cifre in base 64 (ogni cifra corrisponde a 6 bit) codificate utilizzando 64 caratteri ASCII a 7 bit
  - encoded-text è il testo codificato

```
Subject: =?ISO-8859-1?Q?perch=E9_=E8_una_prova?=
```

Encoding Base64

Index	Binary	Char	Index	Binary	Char	Index	Binary	Char	Index	Binary	Char
0	000000	A	16	010000	Q	32	100000	g	48	110000	w
1	000001	B	17	010001	R	33	100001	h	49	110001	x
2	000010	C	18	010010	S	34	100010	i	50	110010	y
3	000011	D	19	010011	T	35	100011	j	51	110011	z
4	000100	E	20	010100	U	36	100100	k	52	110100	0
5	000101	F	21	010101	V	37	100101	l	53	110101	1
6	000110	G	22	010110	W	38	100110	m	54	110110	2
7	000111	H	23	010111	X	39	100111	n	55	110111	3
8	001000	I	24	011000	Y	40	101000	o	56	111000	4
9	001001	J	25	011001	Z	41	101001	p	57	111001	5
10	001010	K	26	011010	a	42	101010	q	58	111010	6
11	001011	L	27	011011	b	43	101011	r	59	111011	7
12	001100	M	28	011100	c	44	101100	s	60	111100	8
13	001101	N	29	011101	d	45	101101	t	61	111101	9
14	001110	O	30	011110	e	46	101110	u	62	111110	+
15	001111	P	31	011111	f	47	101111	v	63	111111	/
Padding		=									



- Cifre in base 64 e loro codifica con caratteri ASCII
- 3 byte → 4 byte (4 caratteri ASCII)
  - +33% di dimensione

Unicode

- Lo standard Unicode ha lo scopo di associare un codice univoco ad “ogni” carattere (www.unicode.org)
  - l’associazione è indipendente dalla lingua, piattaforma, applicativo
  - Definisce la codifica di tutti i caratteri utilizzati nelle principali lingue scritte compresi caratteri di punteggiatura, simboli matematici, simboli tecnici, ecc.
    - La versione 6.0 include 109.449 caratteri
  - supporta tre differenti codifiche
    - UTF-8 prevede che tutti i caratteri siano codificati con una sequenza di byte di lunghezza variabile (da 1 a 4). Ha il vantaggio che codifica direttamente su 8 bit il set ASCII ed è la codifica più usata nel Web
    - UTF-16 codifica su 16 bit (2 byte) i caratteri più usati, mentre gli altri necessitano di 2 unità da 16 bit
    - UTF-32 è la codifica a lunghezza fissa. Ogni carattere Unicode è codificato su 32 bit

UTF-8

- Prevede una codifica con lunghezza variabile da 1 a 4 byte per i caratteri fino a U+10FFFF (RFC 3629)
  - Non codifica tutti i caratteri (code points) in Unicode, ma tutti quelli di maggiore interesse
    - i codici a 1 byte codificano i caratteri ASCII a 7 bit
    - i successivi 1920 caratteri sono codificati con 2 byte e comprendono tutti i caratteri latini, greci, cirillici, arabi, copti....
    - 3 byte sono necessari per codificare tutti i caratteri del Basic Multilingual Plane

bits	Last code point	byte 1	byte 2	byte 3	byte 4
7	U+007F	0xxxxxxx			
11	U+07FF	110xxxxx	10xxxxxx		
16	U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	
21	U+1FFFFF	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

€ U+20AC code: 00100000 10101100 UTF-8: 11100010 10000010 10101100

MIME

- Multipurpose Internet Mail Extensions (RFC 2045-2046)
  - supporto per codifiche dei testi non ASCII 7 bit
  - possibilità di aggiungere allegati (attachment) ai messaggi email
  - gestione di messaggi con più parti (multipart encoding)
  - uso di codifica non ASCII nell'intestazione
  - Aggiunge dei campi di intestazione per definire la struttura del corpo del messaggio (è gestita dall'agente utente) e il tipo di contenuto
    - Mime-Version:
    - Content-Type:
    - Content-Transfer-Encoding:
    - Content-ID:
    - Content-Description:
    - Content-Disposition:

MIME Content-Type

- **Content-Type**
  - Definisce il tipo di dati (Internet Media Type) come tipo/sottotipo
    - `Content-Type: text/plain ; charset=ISO-8859-1`
      - testo non formattato con codifica dei caratteri ISO-8859-1
    - `Content-Type: text/html`
      - un testo con comandi di formattazione HTML
    - `Content-Type: image/jpeg`
      - un'immagine in formato jpeg
    - `Content-Type: multipart/mixed ; boundary="start-separator"`
      - il contenuto è organizzato in più parti di tipo diverso che iniziano con la sequenza `--start-separator`
      - ciascuna parte è descritta da una propria intestazione con la stessa modalità
  - L'informazione sul Content-Type è fondamentale per selezionare il modulo di visualizzazione corretto da parte dell'agente utente

MIME Content-Transfer-Encoding

- Permette di indicare se è stata applicata qualche codifica al tipo dati specificato in Content-Type
  - indica se c'è stata una codifica binary-to-text specificando quale
    - `7bit`
      - è il valore di default che prevede la presenza solo di caratteri ASCII a 7 bit organizzati in linee di massimo 998 byte terminate con CR LF
    - `quoted-printable`
      - usa la codifica quoted-printable per usare solo caratteri ASCII a 7bit. E' adatta per testi in cui i caratteri con codice > 127 sono "pochi" dato che questi vengono codificati con 3 byte (es. =A3)
    - `base64`
      - usa la codifica base64 che è adatta per dati non testuali (i byte con valori > 127 e < 127 sono "equiprobabili")
    - `8bit`
      - caratteri codificati su 8 bit con linee di massimo 998 byte terminate con CR LF
    - `binary`
      - qualsiasi sequenza di byte

Esempio MIME (attachment)

```
MIME-Version: 1.0
Content-type: multipart/mixed;
     Boundary="0__=88256A50008141558f9e8a93df938690918c88256A5000814155"
Content-Disposition: inline

--0__=88256A50008141558f9e8a93df938690918c88256A5000814155
Content-type: text/plain; charset=Windows-1252
Content-transfer-encoding: quoted-printable

Dear Author:

Congratulations! We have been notified that your paper has been accepte=
d

--0__=88256A50008141558f9e8a93df938690918c88256A5000814155
Content-type: application/pdf;
     name="=?Windows-1252?Q?icdar01=5Fcopyright=5Fform.pdf?="
Content-Disposition: attachment; filename="=?Windows-1252?Q?icdar01=5Fcopyright=5Fform.pdf?="
Content-transfer-encoding: base64

JVBERi0xLjIgDSXi48/TDQogDTEwIDAgb2JqDTw8DS9MZW5ndGggMTEgMCBSDS9GaWx0ZXIgL0Zs
YXRlRGVjb2RlIA0+Pg1zdHJlYW0NCkiJrFfZctvIFf0C/UPXvESuUBAaO+YplEx5WDNjuSi6nFTp
```

SMTP

- Simple Mail Transfer Protocol (RFC 821 - RFC 5321)
  - Il protocollo originale è del 1984, l'Extended SMTP del 2008
  - Definisce il protocollo di comunicazione fra due agenti per il trasferimento di un messaggio email
  - Utilizza TCP sulla porta server 25
  - Realizza una modalità push (il messaggio è trasferito dal mittente verso il ricevente quando è disponibile)
  - Prevede una sequenza di comandi (inviati in ASCII) necessaria per il trasferimento dei messaggi
    - HELO <host> - "Saluta" il server
    - MAIL From: <indirizzo> - Indica il mittente del messaggio
    - RCPT To: <indirizzo> - Indica il destinatario (recipient)
    - DATA - Invio corpo del messaggio terminato da un . su una linea
    - QUIT - Chiude la connessione

SMTP session & transactions

- Un sessione SMTP prevede la seguente procedura
  - Un SMTP client (initiating agent) apre una connessione TCP verso il server SMTP
  - Viene aperta una sessione con la negoziazione di parametri
    - il client invia il comando **`HELO`** (**`EHLO`**) al server e attende una lista di opzioni (es. `250 8BITMIME`) che poi vengono attivate o meno
  - Il client effettua una serie di transazioni per inviare uno o più messaggi
    - La transazione consiste di tre sequenze comando/risposta
      - **`MAIL`** - definisce l'indirizzo del mittente
      - **`RCPT`** - definisce il destinatario
      - **`DATA`** - invia il testo del messaggio (header/CR+LF/corpo)
    - In seguito ad ogni comando il server invia una stringa di risultato con un codice che indica se l'operazione ha avuto successo o meno

Un esempio di dialogo SMTP

```
maggini@mcculloch.ing.unifi.it... Connecting to alpha.dii.unisi.it. via relay...
220 alpha.dii.unisi.it ESMTP Postfix (Postfix-19991231) (Linux-Mandrake)
>>> EHLO ultra3.dii.unisi.it
250-alpha.dii.unisi.it
250-PIPELINING
250-SIZE 10240000
250-ETRN
250 8BITMIME
>>> MAIL From:<maggini@ultra3.dii.unisi.it> SIZE=20
250 Ok
>>> RCPT To:<maggini@mcculloch.ing.unifi.it>
250 Ok
>>> DATA
354 End data with <CR><LF>.<CR><LF>
>>> .
250 Ok: queued as BE37A18337
maggini@mcculloch.ing.unifi.it... Sent (Ok: queued as BE37A18337)
Closing connection to alpha.dii.unisi.it.
>>> QUIT
221 Bye
```

Post Office Protocol 3
Internet Mail Access Protocol

- La/Le mailbox sono in remoto
  - Per l'accesso alle mailbox si usa un'applicazione di tipo client/server
  - I protocolli per l'accesso alle mailbox usano una modalità pull
  - POP3 è semplice e ha funzionalità limitate
    - apre una connessione TCP sulla porta 110
    - autentica l'agente con username/password in chiaro (comandi `user`, `pass`)
    - effettua una transazione per recuperare i messaggi eventualmente marcandoli per cancellazione (comandi `list`, `retr`, `dele`)
    - applica l'aggiornamento sulla mailbox al momento del quit
  - IMAP permette la gestione di cartelle remote