Free software per web caching

Laura Abba, Marina Buzzi, Francesco Gennai,
IAT- CNR Pisa

Massimo Ianigro
Area della Ricerca CNR Bari

Abstract

Questo articolo introduce la tecnologia di web caching (concetti base, protocolli, stato dell'arte) e descrive *squid*, un software free sviluppato dai laboratori americani del NLANR (National Laboratory for Applied Network Research, http://www.nlanr.net/), che implementa un cache server dalle caratteristiche interessanti. Nell'ultima parte del lavoro, viene descritto un esempio di web caching a livello nazionale, implementato utilizzando più server *squid* cooperanti in una struttura gerarchica.

1. Web caching

La crescente esigenza di servizi Internet ed in particolar modo le richieste di oggetti da server web (pagine web, file, etc.) contribuiscono all'aumento del traffico di rete, determinando in alcuni casi il congestionamento delle linee e il conseguente degrado nelle prestazioni.

Esistono due differenti tipi di approccio applicabili al problema del traffico web: *web replication*, prevalentemente diffuso negli Stati Uniti, e *web caching*, largamente utilizzato in Europa.

La tecnica di w*eb replication* replica i dati e i servizi disponibili in un particolare sito su server dislocati in differenti aree geografiche, con l'obiettivo di ridurre il traffico indirizzato a server molto acceduti, diminuire il carico computazionale della macchina che offre tali servizi e dare all'utenza un servizio migliore consentendo l'accesso a delle repliche dotate di una migliore connessione telematica. Questo approccio è sostanzialmente analogo a quello seguito per i siti 'FTP', in cui si creano dei duplicati (mirror) dei siti più acceduti in modo tale da consentire agli utenti della rete la disponibilità delle stesse informazioni su macchine meno congestionate o con una migliore connettività.

Il termine *web caching* indica invece la memorizzazione automatica (caching) di pagine, file o, più in generale, oggetti richiesti dai browser (risposte http) in siti più vicini al richiedente rispetto alla sorgente dell'informazione (origin web server).

Tali richieste, originate dagli utenti mediante browser web, vengono trasmesse al *web cache* che verifica la

disponibilità dell'oggetto all'interno della propria memoria o dei dischi (*cache*) e, se presente, fornisce l'oggetto al client che lo aveva richiesto. In caso di richiesta relativa a un oggetto non presente nella cache, il server provvede a contattare il sito web originario (definito nella URL), recupera l'oggetto e lo restituisce al client, conservandone però una copia nella propria memoria in modo che eventuali richieste future dello stesso oggetto possano essere soddisfatte direttamente dal *cache server* stesso.

Le macchine che offrono un servizio di *web caching* possono essere organizzate in modo da cooperare, definendo eventualmente dei rapporti gerarchici tra di esse, oppure agire in modalità *stand-alone*, evitando ogni forma di interazione con altri *web cache*.

In quest'ultima ipotesi, un cache server autonomo offre minori vantaggi rispetto all'adozione di un insieme di più cache server che cooperano (cache peer, cache hierarchy, cache farm, cache array) per la memorizzazione e il recupero di oggetti (internet object), sia in termini di prestazioni che in termini di numero di oggetti trovati dentro le cache (hit rate), di scalabilità e di tolleranza ai guasti (fail-safe policy).

Una struttura di cache server cooperanti (cache mesh) offre quindi molteplici vantaggi:

1. riduzione del traffico di rete (http, ftp); in relazione alla configurazione dei sistemi e alla natura delle richieste provenienti dai vari browser, il traffico può essere ridotto fino al 40%-50%;
2. miglioramento della qualità del servizio (QoS) offerto all'utente, mediante la riduzione dei tempi di accesso alle informazioni;
3. riduzione delle richieste inviate a Web Server molto "visitati", che risultano pertanto gravati da un minor numero di richieste da esaudire;
4. maggior stabilità del servizio, poiché la realizzazione di una rete di cache server cooperanti evita situazioni che presentino un "single point of failure" per quanto concerne l'accesso a Internet;
5. scalabilità della struttura di *web* caching mediante la definizione di nuovi livelli di gerarchia, l'inserimento di nuovi server o la riconfigurazione dei ruoli di cooperazione tra i *cache* server in risposta alle variazioni delle esigenze dell'utenza.

1.1. Cache mesh

Più cache cooperanti sono (logicamente) strutturate in una gerarchia mediante la definizione di relazioni di peering tra cache "vicine" (neighbor). Esistono due tipi di relazioni: parent e sibling.

In una relazione di tipo ‘parent’ viene definita una forma di subordinazione gerarchica tra *cache server* mentre in relazioni di tipo ‘sibling’, i server agiscono allo stesso livello.

Anche le funzionalità implementate sono differenti: i sibling, ricevuta una richiesta, restituiscono l’oggetto solo se esso è memorizzato nella cache (e conserva ancora la sua validità) mentre, in caso di mancanza, non si preoccupano del recupero dell’oggetto stesso.

In una relazione di tipo ‘parent’ invece, il *cache server parent*, nella eventualità che l’informazione richiestagli dal *cache server child* sia assente dalla cache, provvede a richiederla ai suoi cache peer (sibling e parent) o, in mancanza di essi, direttamente al sito web originario [Fig. 1].

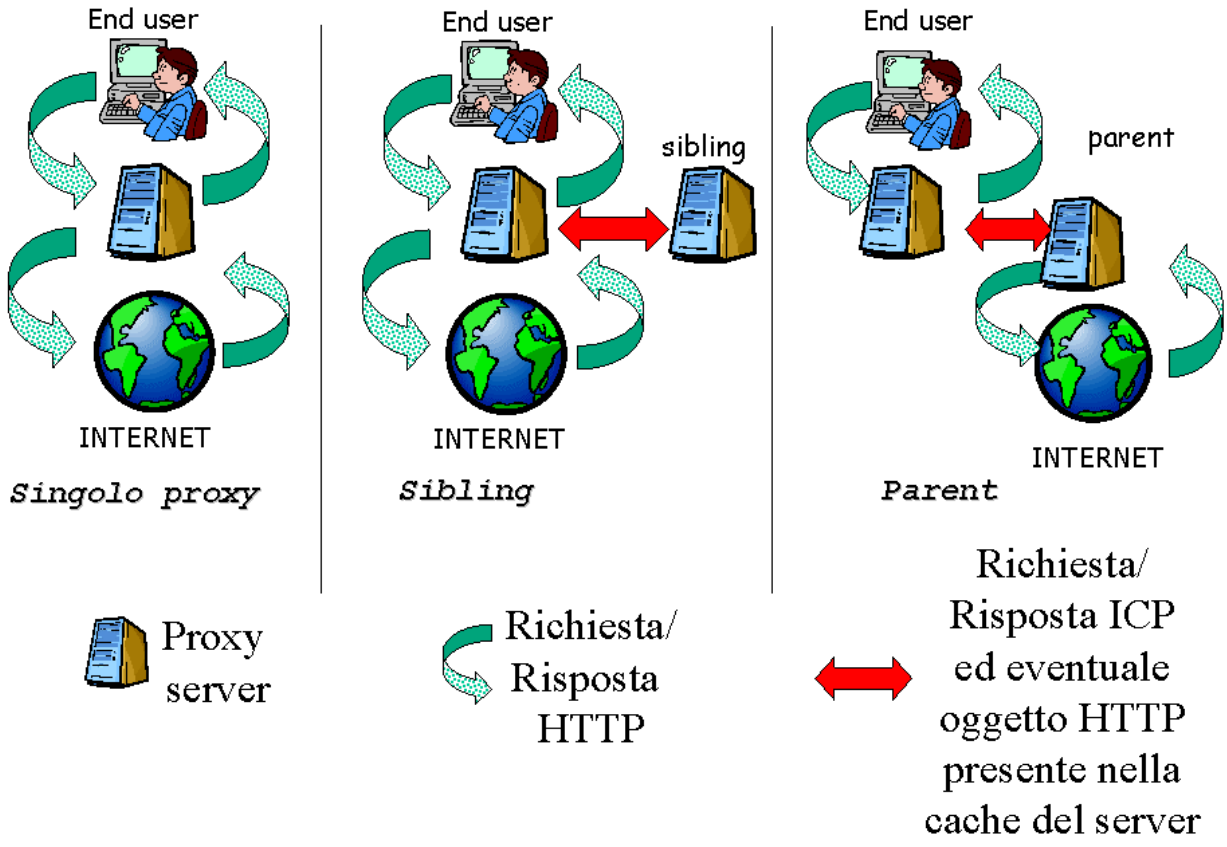

Figura 1 - Schema di relazioni tra *end-user, sibling e parent*

Cache farm o cache array sono esempi di organizzazione di cache non gerarchiche. In questo caso lo spazio delle URL viene suddiviso tra più cache in base ad algoritmi di distribuzione (funzioni hash) in modo da

ottimizzare le prestazioni del sistema e pertanto le richieste originate dai client vengono ripartite in modo uniforme tra più macchine

Nella definizione di una gerarchia di cache server cooperanti è importante tener conto di alcuni fattori:

- più di tre livelli di cache non sono consigliati (a livello nazionale);
- i cache server di livello più alto (top cache) dovrebbero essere collocati "vicini" ai link nazionali/internazionali;
- nella definizione delle relazioni tra cache è necessario tener conto delle politiche di routing (per ottimizzare i flussi dei dati e non creare colli di bottiglia);
- cache server di alto livello devono disporre di adeguate risorse al fine di poter fronteggiare l'elevato carico computazionale che deriva dal loro ruolo primario.

2. I protocolli

Web caching e web replication hanno assunto una importanza fondamentale come dimostra la creazione, all'interno dell'IETF (Internet Engineering Task Force), del gruppo di lavoro WREC (Web Replication and Caching) dedicato allo studio di tali problematiche. Lo scopo del gruppo è studiare l'attuale sviluppo di web caching e web replication, sottomettere protocolli e standard non pubblicamente documentati all'IESG (Internet Engineering Steering Group) per la pubblicazione in forma di RFC (Request For Comments) e fornire raccomandazioni per l'eventuale sviluppo di nuovi protocolli.

Anche se la replica dei servizi web e il caching di risposte http sono paradigmi distinti che richiedono soluzioni differenti, essi devono essere in grado di poter coesistere. Parte del lavoro del gruppo è definire le modalità con cui repliche e cache server possono essere automaticamente scoperti e utilizzati da client.

Il gruppo ha già prodotto un documento [1] che fornisce una tassonomia dei correnti sviluppi di caching e replication e definisce la terminologia di riferimento. Sono stati inoltre pubblicati altri draft [2], [5], [8], [9]. Tutta la documentazione relativa alle attività del gruppo (draft, RFC, archivi della lista di discussione operante all'interno del gruppo, etc.) è accessibile alla URL: http://www.wrec.org/.

Nel seguito saranno brevemente introdotti i protocolli di interazione tra cache. Per i dettagli di funzionamento dei protocolli e la definizione del formato delle informazioni, si rimanda alla bibliografia allegata.

2.1. ICP

Internet Cache Protocol [2], [3], [4] è un protocollo di comunicazione tra cache. E' utilizzato per determinare la presenza di oggetti nelle cache vicine. Le cache, scambiandosi messaggi ICP, raccolgono informazioni che utilizzano per determinare la locazione più appropriata da cui recuperare un oggetto. Il protocollo ICP si basa sul protocollo UDP (User Datagram Protocol) che non garantisce l'affidabilità della comunicazione (UDP è un protocollo *connectionless*). La perdita di pacchetti ICP può pertanto essere utilizzata per fornire una stima della congestione della rete e sulla disponibilità dei server. Il problema di questo protocollo è la mancanza di informazioni sull'header HTTP associato a un oggetto, che contiene informazioni vitali per l'accettazione o il rifiuto dell'oggetto stesso. Il protocollo, disponibile nella versione 2, è largamente implementato sia in prodotti free che commerciali.

2.2. HTCP

Hyper Text Caching Protocol [5] è un protocollo di comunicazione inter-cache nato dal desiderio di colmare le lacune di ICP. HTCP, a differenza di ICP, include nel pacchetto l'header HTTP che contiene informazioni importantissime per le cache. Infatti uno dei maggiori cambiamenti nel passaggio dal protocollo HTTP/1.0 a HTTP/1.1 è l'aggiunta del supporto per web caching. Gli header abilitano i fornitori delle informazioni a poter specificare direttive per le cache [2].

Utilizzando HTCP è inoltre possibile scoprire cache e dati in esse memorizzati, gestire insiemi di cache e monitorarne le attività. Attualmente il protocollo è supportato da un numero limitato di prodotti.

2.3. Cache Digest

Il meccanismo di Cache Digest [6] affronta il problema della latenza e della congestione associati ai protocolli ICP e HTCP. A differenza di tali protocolli, Cache Digest supporta peering tra cache server senza richiedere uno scambio richiesta-risposta, consentendo ai server di mantenere un sommario del contenuto delle cache (Digest) con cui essi hanno relazioni. I server ad intervalli regolari si scambiano i Cache Digest.

Utilizzando un Cache Digest è possibile determinare con grado di accuratezza relativamente alto se l'oggetto corrispondente a una certa URL è memorizzato all'interno di un server. Questa funzionalità è ottenuta dando in input ad una funzione hash la URL richiesta e il metodo http corrispondente, e confrontando il risultato ottenuto con i contenuti dei Cache Digest.

Questa tecnica consente quindi di rendere disponibile in formato compatto un sommario del contenuto di un server riducendo fortemente le comunicazioni tra cache.

2.4. CARP

Cache Array Routing Protocol [7] consente di realizzare un cluster di proxy web cache. Utilizzando una funzione hash, CARP consente di dividere lo spazio delle URL tra un insieme di cache. Il protocollo include la definizione di una Proxy Array Membership Table, e le regole per scaricare queste informazioni.

Un client http può quindi instradare in modo bilanciato le richieste di URL a tutti i membri del proxy array e, utilizzando l'ordinamento e la suddivisione delle richieste, viene eliminata (o ridotta) la duplicazione dei contenuti delle cache, con un possibile miglioramento nella velocità dei cache hit.

I client possono utilizzare CARP direttamente come funzione hash basata sul meccanismo di selezione dei proxy.

2.5. WCCP

L'evoluzione della tecnologia di web caching consente oggi di realizzare soluzioni scalabili e affidabili in modo completamente trasparente all'utente (*transparent caching*). Il traffico web viene automaticamente intercettato e rediretto verso una o più web cache. L'utente non deve più configurare il suo browser (per l'utilizzo di un proxy web cache) né in maniera manuale, né in modalità automatica utilizzando un proxy auto-configuration file (http://home.netscape.com/eng/mozilla/2.0/relnotes/demo/proxy-live.html).

Le funzionalità di *transparent caching* possono essere ottenute a livello di switch o a livello di router.

Il protocollo WCCP (Web Cache Coordination Protocol) [8] è utilizzato per associare un singolo router con una o più web-cache (cache farm) allo scopo di ridirezionare in modo trasparente il traffico HTTP. Il protocollo permette ad una delle cache (designated web-cache) di decidere come il router debba distribuire il traffico ridiretto tra le web-cache associate (per un bilanciamento del carico).

Il router, analizzando i pacchetti che lo attraversano, reindirizza quelli diretti alla porta 80 verso una delle cache associate. In caso di guasto di uno dei cache server, il router provvede automaticamente a disattivare l'inoltro delle richieste verso tale macchina utilizzando i server rimanenti o, nel caso di un singolo server, inviando i dati direttamente verso il sito www (fail-safe policy).

La comunicazione tra cache rimane invariata: esse possono continuare ad interagire utilizzando protocolli standard quali ICP e HTCP.

La versione 1.0 del protocollo, definito dalla CISCO, è stata resa di pubblico dominio in forma di draft [8], in modo da consentire il supporto del WCCP anche in prodotti freeware o di altre aziende. La versione 2.0, che introduce nuove ed interessanti estensioni al protocollo, è ancora soluzione proprietaria CISCO.

2.6. Protocolli multicast

I protocolli basati su trasmissioni multicast non sono molto diffusi anche perchè richiedono l'utilizzo di reti che supportino tali servizi (come M-BONE). A questa motivazione si aggiungono ulteriori considerazioni: le infrastrutture, le configurazioni, i tunnel multicast sono spesso instabili con possibile perdità di connettività; l'utilizzo di trasmissione multicast riduce il numero di ICP query, ma non il numero di risposte perciò i benefici sono relativi; dato che non ci sono particolari requisiti per unirsi ad un gruppo multicast, l'utilizzo di trasmissioni multicast può esporre la cache a problemi di privacy: chiunque può unirsi a un gruppo e cercare di intercettare i pacchetti ICP [10].

Il protocollo ICP implementa il supporto per reti multicast, anche se questa funzionalità è ad oggi scarsamente utilizzata.

Nell'ambito del progetto LSAM (Large-Scale Active Middleware) è stata sviluppata una estensione del proxy cache Apache (LSAM Proxy Cache) che usa push multicast di pagine correlate, basate su gruppi di interesse automaticamente selezionati, in modo da caricare le cache nei loro naturali punti di aggregazione di rete (http://www.isi.edu/~lsam/).

Lo scopo del progetto Adaptive Web Caching (AWC) è invece progettare e implementare protocolli per supportare un sistema di web caching globalmente distribuito, auto-configurante e altamente adattativo (maggiori informazioni sono disponibili alla URL http://irl.cs.ucla.edu/AWC/).

3. Squid

3.1. Caratteristiche

Squid è un high-performance proxy cache server che supporta il caching di oggetti FTP, gopher, e HTTP [10]. Il codice è scritto in C ed i sorgenti, liberamente distribuiti, (accessibili dalla URL

http://squid.nlanr.net/Squid/) evolvono principalmente grazie al lavoro dedicato dai laboratori del NLANR, ma, in parte, anche con il contributo di tutta la comunità scientifica internazionale.

Squid è multipiattaforma: è disponibile per pressoché tutte le versioni di Unix (AIX, FreeBSD, HP-UX, Linux, OSF/1, Solaris, SunOS, etc.) e per OS/2. E' possibile compilare e utilizzare *squid* anche su sistemi Windows NT, ma allo stato attuale il software non è ancora completamente affidabile su tale piattaforma. *Squid* supporta completamente i protocolli ICPv2 e Cache Digest e supporta parzialmente HTCP e CARP. E' già cominciata la fase di sviluppo per includere il supporto del protocollo WCCP (vedi http://www.ircache.net/todo/). E' comunque già possibile utilizzare *squid* in soluzioni di transparent caching [10].

Squid è un prodotto molto flessibile che offre la possibilità di configurare una serie di funzionalità molto interessanti, inclusa la possibilità di utilizzarlo come httpd-accelerator per ridurre il carico del server web e della rete locale (http://squid.nlanr.net/Squid/FAQ/FAQ-20.html#what-is-httpd-accelerator).

La distribuzione del prodotto consiste in un server primario: *squid*, un programma per il DNS lookup: *dnsserver*, dei programmi opzionali (riscrittura delle richieste, attivazione dell'autenticazione, etc.) e degli strumenti di gestione. In particolare il prodotto offre la possibilità di gestione remota via WEB (mediante moduli di tipo CGI) o via SNMP (Simple Network Management Protocol).

Squid supporta SSL, può essere configurato per lavorare dietro a un firewall, offre un estensivo controllo degli accessi e mantiene il log dell'intera richiesta.

3.2. Supporto

Nell'utilizzo di prodotti free, una delle maggiori preoccupazioni è legata al supporto. I prodotti infatti sono rilasciati senza alcun onere monetario e di solito non prevedono il supporto agli utenti, la presenza di una documentazione accurata e la manutenzione del prodotto nel tempo. In prodotti largamente utilizzati, come *squid*, questo problema è meno sentito, poiché vi sono società che commercializzano sia il supporto all'utenza che software aggiuntivi (come pushing di URL) [10].

Nello specifico *squid* offre una lista di discussione molto attiva squid-users@ircache.net (http://squid.nlanr.net/Squid/mailing-lists.html) che rappresenta un buon supporto per gli amministratori di web cache e si integra agli ausilii tradizionali quali FAQ o documentazione in formato elettronico.

La possibilità di avere supporto e feature addizionali da compagnie private operanti in differenti aree geografiche, ci dà una indicazione sulla tendenza del mercato. La larga diffusione di prodotti free di alta qualità attira gli investimenti delle compagnie e sono numerosi i casi di prodotti nati in ambito freeware e poi evoluti in forma di prodotti commerciali.

3.3. Esempio di utilizzo

L'architettura di cache server attualmente utilizzata nel CNR è costituita da due livelli gerarchici di cache server cooperanti [11]. I server sono omogenei e utilizzano *squid* in diverse versioni e su piattaforme eterogenee. Al primo livello troviamo i proxy cache server di Bologna e Roma, correlati mediante una reciproca relazione di sibling mentre tutti i rimanenti server sono di secondo livello. La figura 2 mostra una semplificazione dell'architettura del CNR. Le reti si attestano su tre Point of Presence (POP) della rete GARR (http://www.garr.it/) allocati a Milano, Roma e Bologna. A breve è anche prevista attestazione delle linee CNR al POP GARR di Napoli (http://www.garr.it/mappagarr/garr-b-mappagarr.shtml).

Ogni server di secondo livello utilizza entrambi i server di primo livello ma con pesi diversi. Questo consente di avere una configurazione che utilizza un server primario e un server di back-up. Nell'algoritmo di selezione del parent sono privilegiati i server con pesi maggiori.

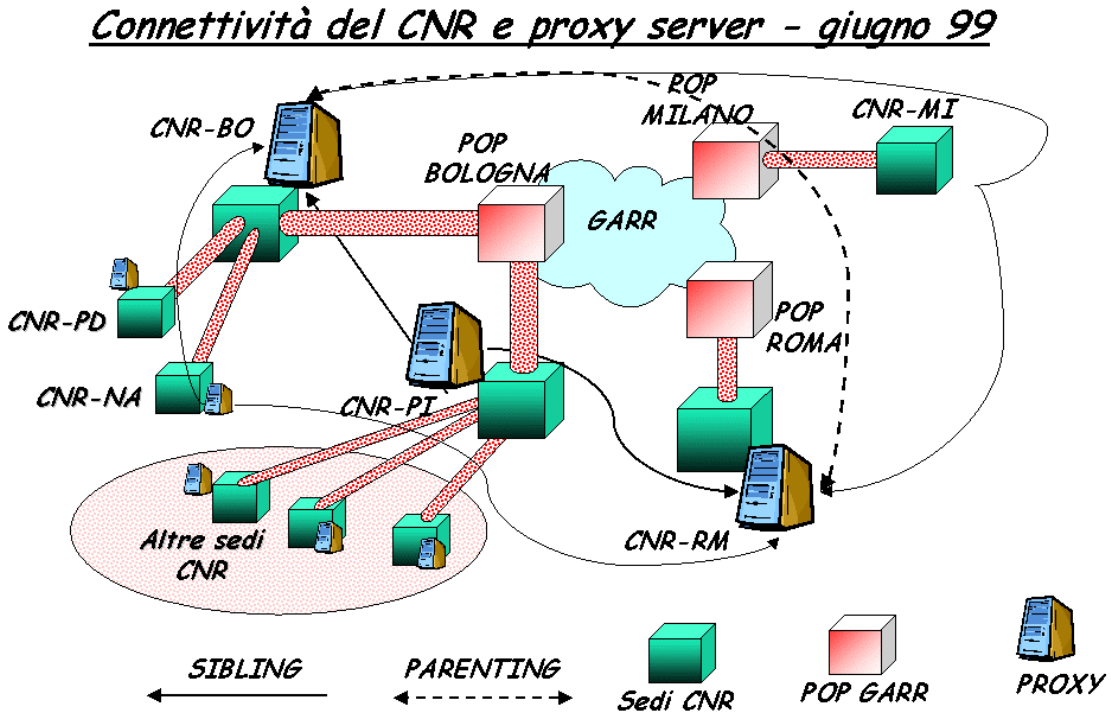

Figura 2 - Schema semplificato dell'architettura di caching del CNR

Dato che in questo momento il collegamento tra Bologna e il CNAF è abbastanza saturo, solo Padova utilizza Bologna come parent primario e Roma come parent di back-up. Tutti gli altri proxy invece utilizzano Roma come server primario e Bologna come parent di back-up. Nella figura le relazioni tra i server di primo e secondo livello sono evidenziate con linea continua (parenting) e linea tratteggiata (sibling).

L'attuale configurazione sarà opportunamente rivista dopo l'attivazione del servizio di web caching della rete GARR , a cui il CNR afferisce.

4. Strumenti di valutazione

Numerosi strumenti di benchmarking possono essere utilizzati sia per testare e mettere a punto i prodotti sia per valutare soluzioni di caching (configurazioni). Ad esempio, Web Polygraph è uno strumento di benchmarking per proxy web cache sviluppato dai laboratori del NLANR. Il prodotto, che include simulatori di client e server web altamente performanti, è disponibile gratuitamente alla URL http://polygraph.ircache.net/.

Polygraph [12] è in grado di modellare una varietà di workload per micro e macro benchmarking. E' stato utilizzato come benchmark per una "competizione" (bake-off) tra sviluppatori di prodotti di web caching (free e proprietari) che si è svolta negli Stati Uniti nel Marzo 1999. I risultati della competizione, molto interessanti, sono disponibili alla URL http://bakeoff.ircache.net/. In questo contesto, tra i prodotti free Peregrine, sviluppato dalla Wisconsin University, sembra avere interessanti caratteristiche [13].

La gestione di cache vicine a link internazionali o nazionali dove confluisce una grande mole di traffico necessità di prodotti che offrano un alto troughput. In queste situazioni *squid* potrebbe generare colli di bottiglia comunque superabili utilizzando più copie del prodotto ma in tal caso andrebbe valutato l'onere di gestione di un grande numero di server rispetto ai benefici che ottenibili.

5. Conclusioni

L'utilizzo di tecniche di web caching consente una significativa riduzione del traffico di rete (e dei costi) e migliora la qualità del servizio offerto all'utente (QoS).

Oggi il mercato offre una vasta scelta di soluzioni che vanno da sofware free (come *Squid* dei laboratori dell'NLANR o *Peregrine* della Wisconsin University) a prodotti commerciali basati sia su piattaforme *hardware* specializzate che di tipo *general-purpose* (CacheFlow, Alteon, Foundry Networks, Network Appliance, Novell, IBM, InfoLibria, Cisco, Entera, Inktomi, …).

Nella progettazione e realizzazione di un servizio di Web Caching dovranno essere pertanto considerati numerosi fattori quali la topologia di rete, il traffico sostenuto, la disponibilità di personale tecnico, le prestazioni della strumentazione utilizzata e i costi.

Un fattore di indubbia importanza nella valutazione di un prodotto è l'adozione di protocolli standard che garantiscono l'interoperabilità con prodotti di altri costruttori.

Bibliografia

[1] Internet Web Replication and Caching Taxonomy - Ingrid Melve, Gary Tomlinson - , Febbraio 1999 http://www.ietf.org/internet-drafts/draft-melve-wrec-taxonomy-00.txt

[2] Inter Cache Communication Protocol – Ingrid Melve - Internet draft, Novembre 1998 - http://www.wrec.org/Drafts/draft-melve-intercache-comproto-00.txt

[3] Internet Cache Protocol (ICP), version 2 - D. Wessels, K. Claffy , Settembre 1997 http://squid.nlanr.net/Squid/rfc2186.txt

[4] Application of Internet Cache Protocol (ICP), version 2 – D. Wessels, K. Claffy - RFC 2187, Settembre 1997 - http://squid.nlanr.net/Squid/rfc2187.txt

[5] Hyper Text Caching Protocol (HTCP/0.0) – Paul Vixie, D. Wessels - Internet draft, Settembre 1998 - http://www.wrec.org/Drafts/draft-vixie-htcp-proto-04.txt

[6] Cache Digest specification, version 5 - Martin Hamilton, Alex Rousskov, Duane Wessels, Dicembre 1998 - http://squid.nlanr.net/Squid/CacheDigest/cache-digest-v5.txt

[7] Cache Array Routing Protocol v1.0 - Vinod Valloppillil, Keith W. Ross - Internet draft, Febbraio 1998 - http://www.ietf.org/internet-drafts/draft-vinod-carp-v1-03.txt

[8] Web Cache Coordination Protocol V1.0 - M Cieslak, D Forster - Internet draft, Giugno 1999 http://www.wrec.org/Drafts/draft-ietf-wrec-web-pro-00.txt

[9] Client-Cache Communication – Ingrid Melve - Internet draft, Novembre 1998 - http://www.wrec.org/Drafts/draft-melve-clientcache-com-00.txt

[10] SQUID Frequently Asked Questions - Duane Wessels - http://squid.nlanr.net/Squid/FAQ/FAQ.html

[11] Il servizio Proxy Cache Server del CNR – M. Buzzi, Febbraio 1999 - http://www.iat.cnr.it/~cachemgr/CacheServer/

[12] Web Polygraph Frequently Answered Questions - Laboratori NLANR, Febbraio 1999 - http://polygraph.ircache.net/FAQ/

[13] Peregrine Web Proxy Software - Pei Cao - http://www.cs.wisc.edu/~cao/peregrine.html