Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Reti Attive: Standard proposti ed ambienti di esecuzione

Daniela Di Fatta, Giampiero Rizzo

RT-ICAR-PA-04-04 **gennaio 2004**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sezione di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Reti Attive: Standard proposti ed ambienti di esecuzione

Daniela Di Fatta[1], **Giampiero Rizzo**[1]

Rapporto Tecnico N. 4 : ***Data:***
RT-ICAR-PA-04-04 **gennaio 2004**

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sezione di Palermo Viale delle Scienze edificio 11 90128 Palermo

INDICE

1 Introduzione

Le reti attive sono reti a commutazione di pacchetto in cui i pacchetti possono contenere frammenti di codice che vengono eseguiti sui nodi intermedi della rete (router). La caratteristica principale che definisce una rete attiva è la possibilità che componenti software degli utenti della rete venga dinamicamente caricato ed eseguito nei nodi di rete. In tal modo, le reti attive forniscono un'interfaccia programmabile dei nodi di rete che permettere la realizzazione di nuovi servizi. Le reti di calcolatori tradizionali applicano il paradigma *store-and-forward*. Una rete consiste di un insieme di nodi collegati da canali trasmissivi. L'unità di base di funzionamento del multiplexing nelle reti tradizionali è il pacchetto, o protocol data unit. Ogni nodo realizza soltanto l'elaborazione necessaria a spedire i pacchetti verso la loro destinazione. I nodi ricevono i pacchetti dagli utenti o da altri nodi, effettuano un calcolo basato sul loro stato interno e sulle informazioni di controllo (intestazione o header del pacchetto) trasportate dentro il pacchetto e, come conseguenza del calcolo, possono spedire uno o più pacchetti verso altri nodi o agli utenti. Le reti permettono che i loro utenti condividono la larghezza di banda della rete come risorsa comune.

Le reti attive cambiano questa visione della rete introducendo il concetto *store-compute-and-forward* (figura 1). I pacchetti attivi trasportano non soltanto i dati ma anche porzioni di codice che saranno eseguite dai nodi intermedi lungo il percorso del pacchetto.

Parecchi vantaggi ma anche parecchie problematiche conseguono da questo nuovo approccio. Un aspetto che forse più degli altri merita attenzione è la sicurezza. Gli svantaggi sono comunque ampiamente compensati dalle nuove possibilità offerte da questa tecnologia. Le reti attive sfruttano tutte le risorse di rete, non soltanto la larghezza di banda, ma anche la capacità di calcolo e di memoria dei nodi estremi e dei nodi intermedi. Una rete attiva permette che i relativi utenti scrivano applicazioni capaci di utilizzare tutte le risorse dei nodi di rete quali CPU, memoria centrale e memoria secondaria dei nodi intermedi.

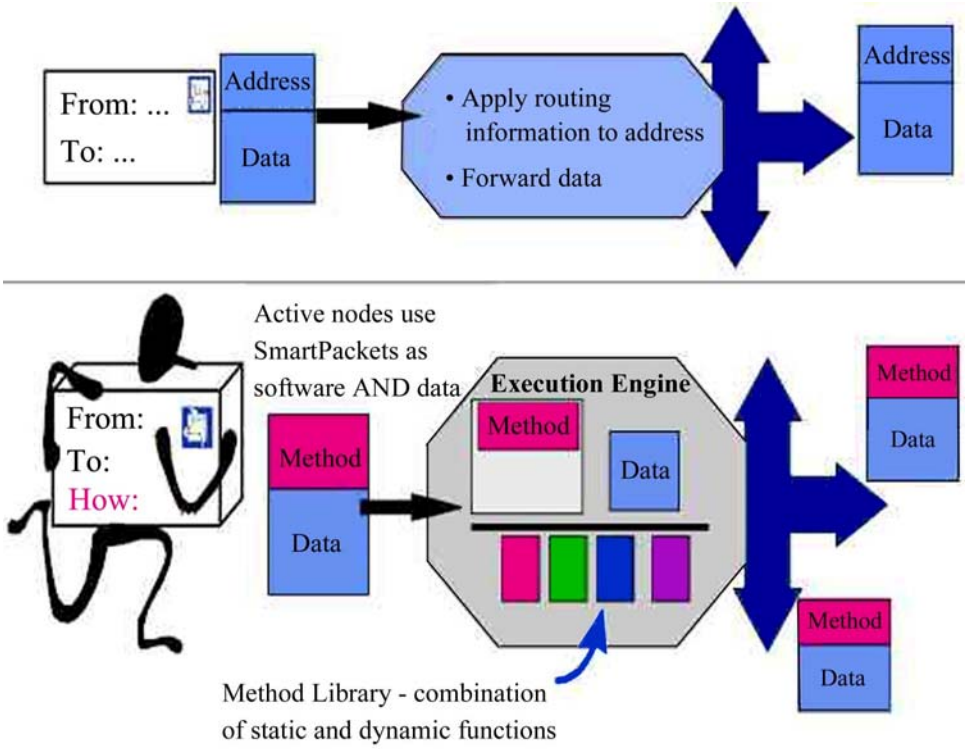

Figura 1: Reti tradizionali a commutazione di pacchetto e Reti Attive (sorgente: [DARPA])

Inoltre, essa fornisce una interfaccia di programmazione (API) per iniettare il codice dell'utente in questi nodi, così permettendo l'adattamento dei protocolli e dei servizi di rete direttamente dall'utente. Le reti attive possono essere considerate come lo sviluppo naturale della attuali reti. I pacchetti tradizionali possono già essere considerati come elementi portanti di piccoli programmi, ma con funzionalità molto limitate; e.g. trasporto dei dati (payload) e specifica del nodo destinazione dove consegnare e valutare il pacchetto. Inoltre la valutazione è costituita soltanto nell'estrazione dei dati. Il carico utile in effetti è costituito dai dati passivi che vengono semplicemente passati ai livelli (protocolli) superiori. Invece, in un ambiente attivo, il pacchetto è costituito da un programma (che naturalmente può contenere anche dati). Il progettista della rete deve così fornire all'utente le API della rete programmabile ed una macchina virtuale adatta ad eseguire i pacchetti attivi nella rete. Un ambiente attivo può essere diviso così in una parte fissa ed una variabile, stabilita dall'utente. Se l'API definisce una macchina completa di Turing, quindi il programmatore non ha virtualmente limiti nella determinazione del comportamento dei nodi; d'altra parte, la parte fissa potrebbe accettare soltanto alcuni parametri predefiniti e solo piccole variazioni da un comportamento standard sono permesse. Nel primo caso può diventare molto difficile comprendere l'influenza dei singoli nodi nel comportamento complessivo della rete, mentre il secondo caso è abbastanza

simile al comportamento dell'attuale Internet, dove le possibili scelte per i parametri sono molto limitate. Un compromesso tra i due approcci è la soluzione più auspicabile.

1.1 Evoluzione, rivoluzione, o involuzione

Una tradizionale architettura di rete è una serie di strati e protocolli, dove ogni layer offre staticamente dei servizi certi allo strato superiore. In questo contesto le reti attive potrebbero essere considerate una rivoluzione. Le reti attive [Smith] differiscono da quelle tradizionali principalmente per quello che non specificano. Invece di definire funzionano i nodi per fornire i servizi di rete, le reti attive stabiliscono degli slot che possono essere insaziati per fornire un particolare tipo di servizio di rete. Questi slot creano un nuovo grado di libertà nell'architettura di rete, il quale apre l'opportunità di accelerare l'evoluzione della rete e quindi favorire nuovi tipi di rete, algoritmi ed applicazioni. In questo modo sarà possibile risolvere un dei più importanti problemi delle attuali reti, che è la difficoltà a scrivere nuovi protocolli.

Attualmente la scrittura di nuovi protocolli richiede tempo, che è quantificabile in anni. L'introduzione delle infrastrutture di rete programmabili permetterà agli utenti di superare questo lungo e noioso percorso che con nuove necessità della rete verrà gestito con la programmazione della rete.

Da un altro lato i puristi del concetti di rete hanno considerato le reti attive un'involuzione. La struttura gerarchica dell'architettura stratificata segue forti principi e permette di far fronte alla progettazione di sistemi complessi. Le reti Attive sembrano infrangere alcune regole fondamentali (ad esempio l'argomento end-to-end [BATTHI]) ed introduce anarchia nella rigorosa organizzazione della struttura stratificata.

La comunità della ricerca sulle reti attive preferisce considerare esse un'evoluzione delle tradizionali architetture di rete "motivandole con tecnologie PUSH e END USER PULL" [TENN)].

I concetti di rete Attiva non sono completamente nuovi. Specifiche soluzioni a molti problemi nell'ambito delle reti sono state proposte e perfino adottate, quali l'esecuzione di calcoli user-driven ai nodi all'interno della rete (esempio i firewalls, web caches e proxies, multicast routers, mobile proxies, e video gateways).

In aggiunta molte delle recenti tendenze delle reti potrebbero essere considerate un sottosistema dell'architettura delle Reti Attive: ad esempio VLAN in PHY layer,

Multiprotocol router, RSVP, RTP, le Application Layer routine sono tutte funzioni che una rete attiva può fornire. Le reti attive possono essere usate come infrastrutture comuni per risolver sia i problemi attuali che quelli a venire senza che necessitino soluzioni ad-hoc.

L'oggetto delle Reti Attive è la costruzione di infrastrutture di rete basate su nodi di rete aperti programmabili dove è possibile distribuire dinamicamente programmi all'interno dei motori dei nodi di rete. La ricerca sulle Reti Attive consiste nello sviluppo di meccanismi per incrementare la flessibilità e l'adattabilità della rete e aumentare il ritmo della distribuzione del software.

2 Possibili utilizzi delle reti attive

2.1 Nuove applicazioni di rete

Lo sviluppo futuro delle reti verso il paradigma attivo dipende fortemente dai benefici reali che possono essere ottenuti dalle sue applicazioni. Si ritiene che molti di questi benefici rientrano nelle seguenti categorie:

- disponibilità di informazioni presso i nodi intermedi,
- capacità di elaborazione lungo il percorso,
- adozione di strategie distribuite.

Disponibilità di informazioni immagazzinate nei nodi intermedi

Agenti mobili possono essere incapsulati e trasportati nel codice attivo delle capsule di una applicazione. Possono accedere ed estrarre informazione contenuta nei nodi intermedi in modo più efficace che con una interazione remota. Per esempio, un agente può utilizzare codice attivo per accedere alla tabella di instradamento di un nodo intermedio e selezionare e filtrare alcune informazioni utili. Infine, può trasmettere tali informazioni all'applicazione remota, o può usarle al fine di prendere decisioni autonomamente dall'applicazione. Molti altri esempi di questo genere possono essere fatti per altre applicazioni di rete, quali, ad esempio, il controllo della congestione, la gestione degli errori, il controllo del traffico, la qualità del servizio (QoS). Un altro esempio molto interessante è quello relativo all'adattamento della funzione di instradamento (routing). Un agente mobile può essere dedicato alla valutazione

del percorso da far intraprendere ai dati dell'applicazione, secondo delle specifiche di QoS dell'utente. Ogni applicazione potrebbe installare nei nodi intermedi la propria politica di controllo, o sfruttare un servizio comune (servizio di default).

Capacità di elaborazione lungo il percorso

Funzioni specifiche delle applicazioni iniettate ed eseguite in un nodo intermedio possono accedere e modificare i dati in transito indirizzati ad altri nodi. Tali modifiche possono dipendere dallo stato attuale della rete o da particolari necessità del nodo destinazione. Le traduzioni del formato dei dati, la compressione con differenti livelli di codifica, codifica e decodifica per motivi di sicurezza e protezione dei dati, sono alcuni degli esempi. La trasmissione multicast (uno-a-molti) è un caso dove questi benefici sono molto evidenti. Le funzioni, iniettate dinamicamente nei nodi intermedi, possono gestire l'arrivo di nuovi utenti, possono modificare dinamicamente l'albero di trasmissione multicast per ottimizzare l'utilizzazione della larghezza di banda complessiva, o, per adattare il formato dei dati alle specifiche differenti degli utenti.

Adozione di strategie distribuite

Applicazioni basate sulle reti attive possono facilmente implementare delle strategie distribuite utilizzando i pacchetti attivi come degli agenti mobili. Esempi di questa potenzialità sono forniti dalle applicazioni già esistenti, quali i web proxy [TENN2], aste e quotazioni di borsa on line [WETH1], firewall distribuiti [TENN2], e la gestione distribuita di alberi di multicast. Un'applicazione particolarmente interessante, proposta in[VV97] è un firewall mobile ad-hoc, il cui scopo è inibire un tipo di "denial-of-service" noto come attacco "SYN-Flooding". Questo attacco consiste nell'inviare un numero tanto elevato di richieste di connessione TCP da occupare tutte le risorse di memoria del server. Per identificare la fonte delle richieste ricevute e per arrestare l'attacco, il server inietta nella rete un agente mobile di difesa capace di riconoscere i pacchetti dell'intruso ed arrestarli nei nodi intermedi sempre più vicino al nodo da cui proviene l'attacco.

2.2 Nuovi servizi di rete

La tecnica di iniettare ed eseguire codice nei nodi intermedi è la chiave della flessibilità di una rete attiva e rende lo sviluppo di nuovi protocolli, servizi e di altre applicazioni di rete più

semplice e potente. Test su nuovi protocolli possono essere effettuati rapidamente su reti reali ed essere sperimentati e non semplicemente simulati. L'aggiornamento del software del dispositivo di rete può essere compiuto a distanza. Nuovi servizi che possonop essere realizzati facilmente in una rete attiva sono l'instradamento dipendente dalla applicazione, già discusso, e l'instradamento parallelo, dove diversi percorsi paralleli sostituiscono il singolo percorso per la trasmissione unicast. Nel processo di sviluppo di nuovi servizi di rete l'architettura di rete attiva consegue cinque obiettivi principali:

Minimizzazione del processo di standardizzazione di protocolli necessari per sviluppare e realizzare servizi end-to-end.

- Riducendo la quantità di consenso generale necessario questo obiettivo copre interessi sia di ricerca che commerciali. I progettisti possono sperimentare con reti reali perché la rete è abbastanza flessibile per soddisfare i loro bisogni durante il funzionamento. Sulla base di uno standard comune, una varietà di nuovi servizi di comunicazione può essere stabilita dinamicamente all'interno di ogni EE.

Aumento della flessibilità nei servizi supportati

- Questo consente di sostenere simultaneamente servizi differenti. Si ha però la necessità degli EE di controllare le risorse del dispositivo di rete per mezzo di un sistema operativo specifico per il nodo attivo. In questo modo l'architettura è simile ad un'architettura in cui molti EE sono simultaneamente presenti.

Interconnessioni di reti operate da diversi operatori

- La sicurezza deve essere una considerazione fondamentale. L'architettura attiva fornisce i servizi di sicurezza agli EE sfruttando le caratteristiche del S.O. del nodo.

Dimensioni e velocità vengono supportate in modo scalabile

- L'elaborazione tradizionale è più veloce e può essere adottata per quei pacchetti che non hanno bisogno di servizi forniti dalle reti attive.

Capacità di comprendere facilmente i protocolli attuali come casi particolari.

- In generale in un'architettura attiva, la pila dei protocolli IP può essere vista come un particolare ambiente di esecuzione, anche se con una API ed una funzionalità molto semplici.

3 Standard Proposti

IEEE P1520 (http://www.ieee-pin.org/)

Parecchie aziende ed ambienti accademici hanno aderito al progetto di costituzione di uno standard, noto come IEEE P1520. Questo progetto vede le reti di telecomunicazioni di domani come un enorme calcolatore - una macchina completamente programmabile che trasporta voce, dati e servizi avanzati per il video. Lo standard proposto specifica in un Interface Definition Language, un insieme di interfacce di programmazione per l'accesso distribuito alle funzionalità di commutazione delle entità di controllo del servizio. I tipi di unità di commutazione considerate in questo standard includono gli switch Asynchronous Transfer Mode (ATM), gli switch che operano con sistema N. 7 (SS7) ed i router IP.

ANEP (http://www.cis.upenn.edu/~switchware/ANEP/)

Lo Active Network Encapsulation Protocol [ANEP] [ANIP6] è un tentativo di fornire interoperabilità fra diversi EE. Nella terminologia di ANEP, un pacchetto consiste di un'intestazione ANEP e un carico utile. L'intestazione ANEP include un campo di contrassegno del tipo, il cui valore rappresenta uno specifico EE (attualmente questa assegnazione è gestita dal Active Network Assigned Number Autority). Se un EE particolare è presente in un nodo, i pacchetti che contengono un'intestazione valida di ANEP con l'identificazione di tipo saranno diretti al EE appropriato.

DARPA Draft (http://www.darpa.mil/ito/research/anets/Arcdocs.html)

Nel programma DARPA vari gruppi di lavoro sono stati formati per discutere il disegno e lo sviluppo di componenti per lo sviluppo di reti attive. Ogni gruppo di lavoro sta elaborando un documento con le discussioni ed i contributi dalla Comunità di ricerca e sono disponibili al pubblico come "ARFCs": Riguardano quattro argomenti: architettura [ARCH], OS di nodo attivo [NODEOS], servizi componibili [CSAN98] e aspetti relativi alla sicurezza [SEC98].

4 Ambienti di Esecuzione (Execution Environments)

In questa sezione prima viene presentata una classificazione degli ambienti di esecuzione delle reti attive a successivamente una rassegna dei progetti di ricerca principali nelle università e nelle industrie sugli ambienti di esecuzioni per reti attive.

4.1 I due approcci principali

Ci sono principalmente due approcci possibili per realizzare le reti attive: un approccio "discreto" e quello "integrato". L'approccio discreto è anche denominato approccio del "nodo programmabile" perché i programmi sono iniettati nel nodo attivo programmabile in pacchetti diversi dai pacchetti dati che tradizionalmente attraversano la rete. L'utente trasmette il programma al nodo di rete (router), dove viene prima immagazzinato e successivamente eseguito quando i dati arrivano al nodo, elaborando quei dati che sono associati al programma. Ai dati occorre associare alcune informazioni che permettono al nodo di decidere quale programma eseguire per gestire il pacchetto dati. Nel metodo integrato, anche chiamato metodo di incapsulamento, un programma viene integrato in alcuni pacchetti dati, che sono quindi denominati capsule. Ogni capsula contiene un frammento di programma che può anche avere alcuni dati inclusi. Quando una capsula arriva nel nodo attivo, un ambiente di esecuzione, "execution environment" (EE), interpreta il programma ed intraprende una certa azione secondo le istruzioni del programma (come la trasmissione dei dati verso la destinazione). In questo metodo, ogni nodo attivo ha un meccanismo interno per caricare il codice incapsulato, un ambiente di esecuzione per eseguire il codice ed un area di memoria temporanea o permanente dove le capsule possono memorizzare e richiamare informazioni. La seguente tabella riassume le caratteristiche principali dei due metodi.

Approccio	Nodo programmabile	Programmi	Iniezione del codice vs trasmissione dati
Discreto	Configurabile	Moduli, switchlet	Out-of-band
Integrato	Ambiente di esecuzione	Programmazione di pacchetto	In-band

Tabella 1: Approccio discreto e integrato

Fino ad oggi diverse architetture sono state proposte per entrambi gli approcci e per soluzioni intermedie (figura 2). Alcune di queste architetture verranno discusse nel seguito di questo documento.

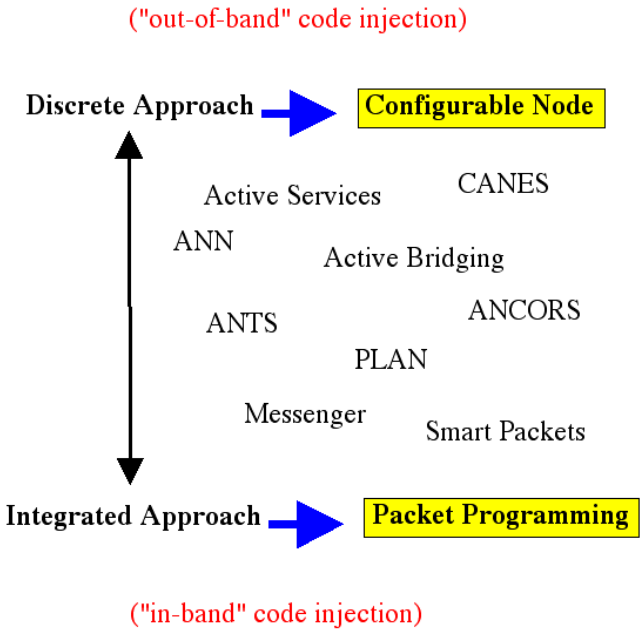

Figura 2: Diversi approcci nelle architetture proposte

4.2 Ambienti di esecuzione

La tabella seguente riassume i vari ambienti di esecuzione per reti attive fino ad ora proposti. Tra questi due rivestono particolare interesse per il nostro laboratorio di reti attive: ANTS e PLAN. Di seguito riportiamo una breve descrizione di ANTS ed una più dettagliata di PLAN.

PROGETTO	PROPONENTE	SITO WEB
ANTS	Massachusetts Institute of Technology, University of Utah University of Washington	http://nms.lcs.mit.edu/activeware/ http://www.cs.utah.edu/flux/janos/software.html http://www.cs.washington.edu/research/networking/ants/
PAN	Massachusetts Institute of Technology,	http://nms.lcs.mit.edu/activeware/
SWITCHWARE and PLAN	Univeristy of Pennsylvania and Bellcore Telcordia	http://www.cis.upenn.edu/~switchware/
SMART PACKETS	BBN Technologies	http://www.ir.bbn.com/projects/spkts/smtpkts-index.html
Messenger	University of Geneva (CH)	http://cui.unige.ch/tios/msgr/home.html
ANN	Washington University Saint Louis	http://www.arl.wustl.edu/arl/projects/ann/ann.html
Liquid Software	University of Arizona	http://www.cs.arizona.edu/liquid/
NETSCRIPT	Columbia University	http://www.cs.columbia.edu/dcc/netscript/
CANEs	Georgia Institute of Technology	http://www.cc.gatech.edu/projects/canes/overview.html
ASP	Stanford Research Institute, Information Science Institute, Metanetworks	http://www.isi.edu/abone/ASP_EE.html
Xbind	Columbia University	http://comet.ctr.columbia.edu/xbind/overview.html
ALPINE	University College London	http://www.cs.ucl.ac.uk/research/alpine/
SAFETYNET	University of Sussex At Brighton	http://www.cogs.susx.ac.uk/projects/safetynet/
ANDROID	British Telecommunications And others	http://www.cs.ucl.ac.uk/research/android/

ANTS

Lo Active Network Transport System (ANTS) è stato uno dei primi sistemi di rete attiva. I pacchetti attivi, o capsule, contengono un riferimento al codice che è necessario eseguire per elaborare il pacchetto stesso. ANTS del MIT utilizza Java come relativo linguaggio di programmazione e la macchina virtuale del Java per l'ambiente runtime. Le caratteristiche di Java rendono ANTS adatto ad una varietà di applicazioni e lo hanno reso particolarmente gradito a molti programmatori. In una fase iniziale ANTS inietta nei nodi del percorso dei pacchetti il codice necessario al protocollo attivo. Successivamente, i pacchetti inviati dalla applicazione possono utilizzare il codice attivo presente sui nodi. A livello più basso della sua

architettura vi sono le capsule di ANTS, che trasportano sia il codice che dati. La rappresentazione risulta molto compatta poiché il codice attivo nei pacchetti viene limitato a riferimenti ad un solo protocollo.

PLAN

Il linguaggio PLAN nasce nell'ambito del progetto SwitchWare alla University of Pennsylvania. SwitchWare è basato su un'architettura a tre livelli:

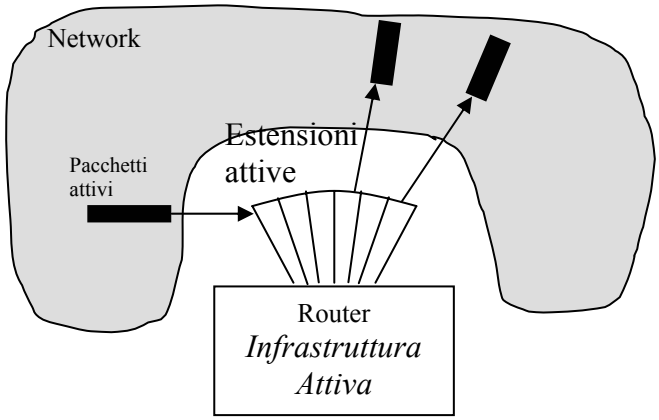

Figura 3: Struttura dell'architettura SwitchWare

Al livello superiore stanno i programmi mobili "leggeri" che sono contenuti in un pacchetto e quindi sono chiamati pacchetti attivi. Tali programmi hanno capacità molto limitate; per esempio non possono modificare in maniera permanente lo stato di un nodo né comunicare con altri pacchetti dal momento che, per ragioni di sicurezza, non sono fidati.

Il secondo livello è quello delle estensioni attive, cioè di programmi OCaml che possono essere caricati dinamicamente nei nodi attivi. Queste estensioni sono impiegate per creare i servizi che possono essere chiamati da dentro i pacchetti.

Il livello più basso, infine, è l'infrastruttura che provvede all'allocazione delle risorse e supervisiona al download degli switchlets, cioè sia dei pacchetti attivi che delle estensioni.

Sono state poi sviluppate le diverse parti del sistema che si occupano di sicurezza e in particolare della rete attiva vera e propria, PLANet. Essa è puramente attiva nel senso che tutti i pacchetti contengono programmi scritti in PLAN e inoltre le funzionalità dei nodi possono essere estese dinamicamente tramite le estensioni attive. Queste scelte progettuali pongono PLANet in una posizione di mezzo lungo due assi ideali che descrivono lo spazio delle implementazioni delle reti attive. Il primo abbraccia i possibili meccanismi per "attivare" una

rete, dalla programmabilità a livello di pacchetto all'estensibilità a livello di nodo. PLANet usa un approccio misto in cui i pacchetti trasportano programmi che possono riferirsi a delle funzioni di uso più generale residenti nel nodo. Il secondo asse comprende invece le possibili posizioni per la valutazione attiva: a un estremo c'è la valutazione nei soli punti finali, all'altro la valutazione ad ogni salto. Di nuovo, PLANet adotta una posizione flessibile che permette la valutazione in alcuni dei nodi intermedi.

La comprensione di PLANet è strettamente legata al linguaggio usato per i pacchetti, cioè PLAN. Esso è basato sul lambda calculus con tipi semplici e fornisce un ristretto insieme di primitive e tipi di dati; include ovviamente le primitive per la valutazione di un'espressione in un nodo remoto con la conseguente creazione di nuovi pacchetti a partire da quelli attualmente sotto valutazione. Un'altra importante caratteristica è la sua semantica che intrinsecamente limita l'uso delle risorse e garantisce sempre la terminazione dei programmi PLAN e che sia loro che i discendenti visitino solo un fissato numero di nodi.

PLAN è stato progettato per essere abbastanza flessibile da scrivere programmi utili, ma abbastanza limitato così che i programmi non pongano rischi per la sicurezza. Per contrasto, le routine di servizio disponibili ai programmi PLAN sono general-purpose e possono richiedere ulteriore protezione tramite crittografia o altri mezzi.

La figura 4 illustra il formato di un pacchetto PLAN.

Poiché, come si è detto, ogni pacchetto contiene un programma PLAN, la standardizzazione del formato di questi pacchetti si riconduce a quella dello schema di codifica di un programma in un pacchetto. Questo semplifica enormemente il compito di un progettista di servizi che si deve occupare solamente di quale informazione deve essere comunicata e non di come ciò debba essere fatto.

Pur volendo portarsi un passo avanti rispetto a Internet, si vuole fare tesoro delle esperienze precedenti che mostrano ad esempio che la maggior parte dei pacchetti richiedono solo un servizio di trasporto essenziale; analogamente in PLANet si assume che l'instradamento e non la valutazione sia la funzione maggiormente richiesta dai pacchetti PLAN. Questa premessa è la giustificazione del campo routFun nel pacchetto. In questo modo si ottiene una strategia flessibile, ma non eccessivamente pesante: quando un pacchetto arriva in un nodo non deve essere necessariamente valutato, ma dal solo esame del campo routFun si deduce quale sia la routine di servizio residente in quel nodo intermedio per determinare il prossimo hop.

Altri campi, come evalDest e source, sono di ovvia interpretazione, mentre una nota a parte merita la terzina che costituisce la chunk (= code hunk). Questa è in effetti il programma trasportato dal pacchetto, ma in realtà è trattato da PLAN come se fossero dei dati che quindi possono essere all'occorrenza frammentati, riassemblati etc. In particolare le chunk possono contenere altre chunk come dati, ottenendo in tal modo un elegante meccanismo di incapsulazione.

Tralasciando altri campi, è infine importante considerare quello denominato rb; esso è simile al time-to-live dell'IPv4 e serve a limitare il consumo di risorse e la persistenza in rete del singolo pacchetto e dell'eventuale progenie.

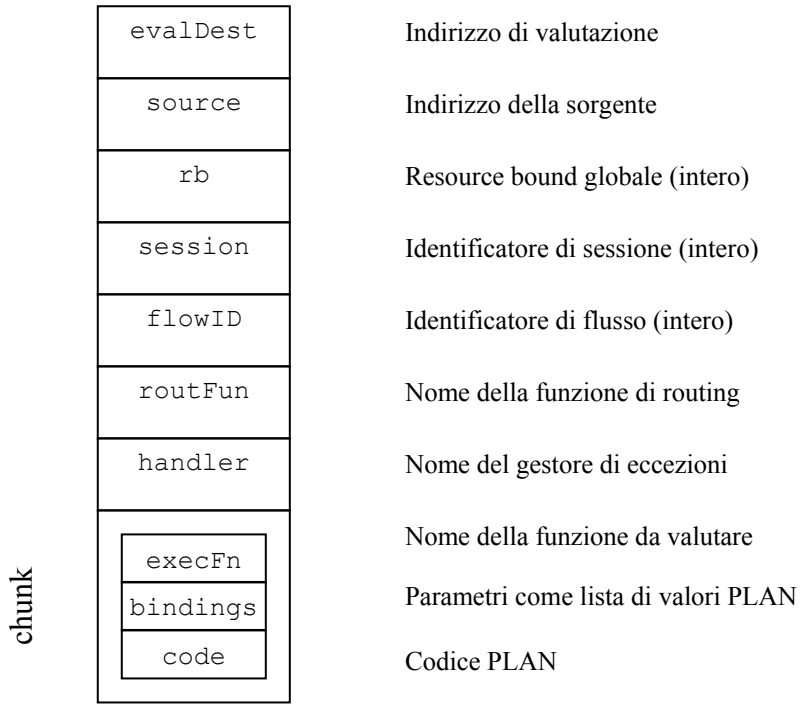

Figura 4: Formato del pacchetto PLAN

5 Bibliografia

[ALE1] Alexander, D.S., Arbaugh, W.A., Keromytis, A.D., Smith, J.M.: Safety and Security of Programmable Network Infrastructures. IEEE Communications Magazine, vol.36, n.10, October 1998, 84 – 92

[ALE2] Alexander, D.S., Arbaugh, W.A., Hicks, M.W., Kakkar, P., Keromytis A.D., Moore, J.T., Gunter, C.A., Nettles, S.M., Smith,J.M.: The SwitchWare Active Network Architecture. IEEE Network Special Issue on Active and Controllable Networks, vol. 12, n. 3, May-June 1998, 29 – 36

[ALE3] Alexander, D.S., Arbaugh, W.A, Keromyts, A.D., Smith, J.M.: A Secure Active Network Architecture: Realization in SwitchWare. IEEE Network Special Issue on Active and Controllable Networks, vol. 12, n. 3, May-June 1998, 37 - 45

[ANEP] D. Alexander et al,
Active Network Encapsulation Protocol. Draft, July 1997.
Available at http://www.cis.upenn.edu/ switchware/ANEP/.

[ARCH] Architectural Framework for Active Networks
Active Network Working Group, Draft, July 27, 1999

[CALV] Calvert, K.L., Bhattacharjee, S., Zegura, E.W., Sterbenz, J.: Directions in Active Networks. IEEE Communications Magazine, vol.36, n.10, October 1998, 72 - 78

[CHEN] Chen, T. M.,: Evolution to the Programmable Internet. IEEE Communications Magazine, vol.38, n. 3, March 2000, 124 – 128

[CLAR90] D. D. Clark and D. L. Tennenhouse, "Architectural Considerations for a New Generation of Protocols", In SIGCOMM '90, 1990.

[DARPA] http://www.darpa.mil/ito/research/anets/

[ENG95] D. R. Engler et al. "Exokernel: An Operating System Architecture for Application-Level Resource Management", In 15th Symp. on Operating Systems Principles, 1995.

[MN00] Moore J. T., Nettles, S, M., Towards Practical Programmable Packets,

[NODEOS] NodeOS Interface Specification
AN Node OS Working Group Draft, January 24, 2000

[NYGREN] Nygren, E., The Design and Implementation of a High Performance Active Network Node, master's thesis, MIT, Cambridge, Mass., 1998

[O'MA92] S. W. O'Malley and L. L. Peterson. "A dynamic network architecture" ACM

Transactions on Computer Systems, 10(2):110-143, May 1992.

[PBAA13] G. Phillips, B. Braden, J. Kann, and B. Lindell.
Writing an Active Application for the ASP Execution Environment (Release 1.3)

[SEC98] Security Architecture Draft.
AN Security Working Group. Draft. 1998

[SMITH] Smith, J. M., Calvert, K.L., Murphy, S. L., Orman, H. K., Peterson, L.L.: Activating Networks: A Progress Report. IEEE Computer, Vol. 32 N. 4, April 1999, 32 – 41

[SCHWAR] Schwartz, B., Jackson, A., Strayer, T., Zhou, W., Rockwell, R., Partridge, C.: Smart packets for Active Networks. Proc. of IEEE 2nd Conference on Open Architectures and Network Programming (OPENARH '99), March 1999

[TENN97] Tennenhouse, D. L., Smith, J.M., Sincoskie, W.D., Wetherall D.J., Minde, G.J.: A Survey of Active Network Research. IEEE Communications Magazine, Vol. 35, No. 1, January 1997, 80-86

[TENN96] Tennenhouse, D. L., Wetherall, D.J.: Towards an Active Network Architecture, Computer Communication Review, Vol. 26, No. 2, April 1996

[WETH1] Wetherall, D.J., Legedza, U., Guttag, J.: Introducing New Internet Services: Why and How. IEEE Network Magazine Special Issue on Active and Programmable Networks, vol. 12, n.3, May-June 1998

[WETH2] Wetherall, D.J., Guttag, J., Tennenhouse, D.L.: ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols. IEEE OPENARCH'98, San Francisco, CA, April 1998