Università degli Studi di Bologna

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica
Reti di Telecomunicazioni LS

Realizzazione di un modello di router ottico in ambiente open source

Tesi di Laurea di:
Raul Cafini

Relatore:
Prof. Ing. Carla Raffaelli

Correlatori:
Dott. Ing. Walter Cerroni
Dott. Ing. Michele Savi

Sessione terza

Anno Accademico 2008/2009

Parole chiave:

Reti Ottiche
Multi-granular switching
Optical Packet Switching
Router programmabile
Click! Modular Router
Linux

D.E.I.S., Dipartimento di Elettronica, Informatica e Sistemistica.
Università di Bologna.

La tesi è scritta in LaTeX 2_ε, utilizzando come testo di riferimento [1].
La stampa è in PostScript.
Le immagini sono create in Adobe® Photoshop® Elements 2.0.

Ai compagni di viaggio,
tesoro più grande
di questa esperienza,
con la speranza di condividere insieme
nuove mete.

Aneddoto

Mi lamentavo con mia madre di quanto fosse difficile
e spaventoso quell'esame all'università.
Lei si inclinò verso di me, mi diede un buffetto sulle spalle e mi disse:
– *Sappiamo bene come ti senti, tesoro, ma ricorda:*
tuo padre alla tua età combatteva contro i tedeschi. –
da *The last lecture.*
Randy Pausch[1].

[1]Randy Pausch (Baltimore, 23 ottobre 1960 – Chesapeake, 25 luglio 2008)

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

Internet non è al passo con i tempi, ma con il futuro.

Anonimo.

Negli ultimi anni nel mondo della comunicazione, ed in particolare nel settore informatico e della rete Internet, lo scenario globale su cui le moderne tecnologie lavorano ogni giorno è in continua evoluzione.

Da un lato la necessità di trasferire una sempre maggiore quantità di informazioni nel minor tempo possibile, dall'altro la crescita esponenziale degli utenti unita alla proliferazione di servizi con richieste in termini di banda e velocità sempre maggiori (ad esempio la fornitura di flussi dati audio e/o video) ha posto il problema tra i gestori della rete, i centri di ricerca e le aziende del settore su come adattare l'infrastrutura esistente ai nuovi requisiti.

In questo senso il trasporto del traffico su tecnologia ottica ha ricoperto un ruolo di primo piano, in quanto è proprio grazie a questa tecnologia se oggi riusciamo a far fronte a questa enorme richiesta in termini di velocità, quantità e qualità nella trasmissione delle informazioni sulla rete.

Anche se le dorsali intercontinentali per telecomunicazioni si basano ormai da alcuni decenni su queste tecnologie, è solo a partire dalla scorsa decade, con l'emergere di questi nuovi scenari, che si è cominciato a sfruttare a pieno le potenzialità delle reti ottiche e a pensare a come trasportarle, grazie anche alla diminuzione dei costi di realizzazione e di posa, in ambienti metropolitani fino a piccole realtà come enti o istituti.

Tutte le problematiche del trasporto delle informazioni nel dominio ottico, il rispetto e la compatibilità verso i sistemi esistenti (ad esempio con le reti ATM o quelle basate su protocollo IP), il processamento delle informazioni di instradamento ed inoltro, i colli di bottiglia e i limiti tecnologici nella realizzazione di alcuni componenti, altrimenti elementari nel dominio elettromagnetico, rappresentano un campo molto attivo della ricerca in questo settore.

Nel seguito introdurremo ed analizzeremo queste problematiche nei loro aspetti di rilievo, forniremo dei cenni sulla struttura e sul funzionamento base dei punti di accesso e smistamento delle informazioni, i *router*, che permettono l'interconnessione di tali reti, e di alcune soluzioni che implementano le tecnologie proposte.

Una volta fornite queste nozioni, introdurremo una architettura per *router* ottici di tipo modulare, sviluppata presso il dipartimento da parte dei correlatori di questo testo, la cui principale caratteristica consiste nell'essere altamente riconfigurabile, in modo da adattarsi a requisiti, anche futuri, come la multi-granularità nella gestione del traffico.

Il lavoro di tesi consisterà dunque nella implementazione software di questa soluzione architetturale mediante l'uso di strumenti di programmazione modulari e altamente configurabili. Dato inoltre l'alto costo dei dispositivi base per la manipolazione delle informazioni su fibra (dai *laser* fino ai convertitori di lunghezza d'onda) il fine ultimo sarà quello di dimostrare la fattibilità della realizzazione di un vero e proprio banco di prova per architetture ottiche, in grado di eseguire, con contenute limitazioni, su un comune *hardware* elettronico e che sia in grado di valutare le prestazioni e i benefici dei sistemi che operano nel dominio della luce, cercando di riprodurre il più fedelmente possibile il comportamento di una matrice di commutazione ottica, contenendo però i costi che derivano dall'uso di veri dispositivi.

Capitolo 1

Le reti ottiche

Le stelle sono piccole fessure attraverso le quali fuoriesce la luce dell'infinito.

Confucio, filosofo cinese.
(551 a.C. - 479 a.C.)

In questo capitolo introdurremo la tecnologia alla base delle comunicazioni nel dominio ottico, analizzandone il principio di funzionamento, la composizione, le tecniche di trasmissione dei dati e le ultime soluzioni l'incremento delle prestazioni e la riduzione dei costi.

1.1 Le fibre ottiche

Le fibre ottiche sono filamenti di materiali vetrosi o polimerici, normalmente disponibili sotto forma di cavi, realizzati in modo da poter condurre la luce. Sono classificate quindi come guide d'onda dielettriche, permettono cioè di convogliare al loro interno un campo elettromagnetico di frequenza sufficientemente alta (in genere in prossimità dell'infrarosso) con perdite estremamente limitate. Sono flessibili, immuni ai disturbi elettrici ed alle condizioni atmosferiche più estreme, e poco sensibili a variazioni di temperatura e per questi aspetti vengono comunemente impiegate nelle telecomunicazioni anche su grandi distanze e nella fornitura di accessi di rete a larga banda (dai 10 Mbit/s al Tbit/s usando le più raffinate tecnologie di multiplazione)[2].

1.1.1 Composizione

Ogni singola fibra ottica è composta da due strati concentrici di materiale trasparente estremamente puro: un nucleo cilindrico centrale, detto *core*, ed

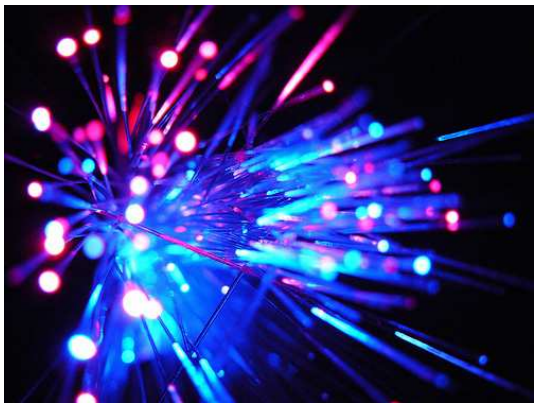

Figura 1.1: Un fascio di fibre ottiche[2].

un mantello o *cladding* attorno ad esso. La fibra ottica funziona come una specie di specchio tubolare, la luce che entra nel *core* ad un certo angolo (detto angolo limite) si propaga mediante una serie di riflessioni sulla superficie di separazione fra i due materiali[2].

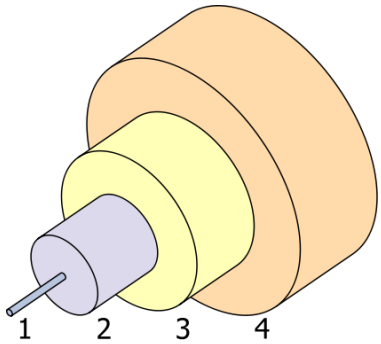

Figura 1.2: Sezione di un cavo in Fibra Ottica: 1 - *Core*, 2 - *Cladding*, 3 - *Buffer*, 4 - *Jacket*[2].

Nel silicio destinato alla produzione del *core* viene aggiunto del germanio in modo da aumentarne l'indice di rifrazione senza variarne l'attenuazione. Il *cladding* invece ha un indice di rifrazione inferiore (ottenuto mediante drogaggio al boro). Intorno a questi due strati vi è un mantello detto *buffer* con uno spessore maggiore della lunghezza di smorzamento dell'onda evanescente, caratteristica della luce trasmessa, in modo da catturare la luce che non viene riflessa nel *core*. Infine un guaina protettiva polimerica detta *jacket* ricopre il tutto per dare resistenza agli stress fisici e alla corrosione ed evitare il contatto fra la fibra e l'ambiente esterno. Le fibre ottiche possono essere realizzate in fibra di vetro (che però dà luogo a fragilità e difficoltà nel caso di raccordi) o in polimeri (se costituita da una materia plastica, in tal caso risulta molto più facile da maneggiare e più resistente allo stress meccanico). Due tratti di fibra

ottica dello stesso tipo possono essere giuntati mediante semplice fusione, che se ben eseguita mediante strumenti di precisione comporta una attenuazione inferiore a 0,05 dB[2].

1.1.2 Principio di funzionamento

Una descrizione approfondita e scientifica del funzionamento delle fibre ottiche richiederebbe nozioni di ottica quantistica ma semplificando e usando un paragone di ottica classica, nelle fibre ottiche avviene un fenomeno di riflessione totale interna, per cui la discontinuità dell'indice di rifrazione tra i materiali del nucleo e del mantello intrappola la radiazione luminosa finché questa mantiene un angolo abbastanza radente, in pratica finché la fibra non compie curve troppo brusche. Infatti quando un raggio luminoso passa da un materiale (ad esempio il silicio fuso) ad un altro (ad esempio l'aria) questo si rifrange (ovvero si curva) sul confine tra i due materiali[2].

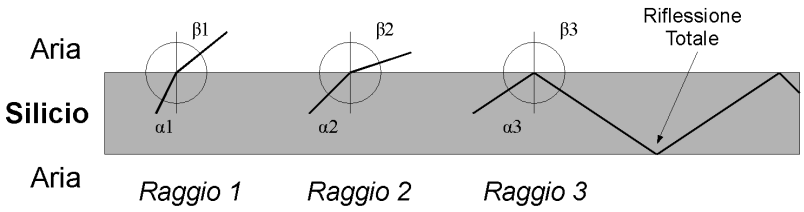

Figura 1.3: Tre esempi di un raggio di luce che dall'interno di una fibra di silicio colpisce il confine aria/silicio con diversi angoli di incidenza, fino all'ottenimento della riflessione totale.

In figura sono mostrati tre esempi di un raggio di luce che dall'interno di una fibra colpisce il confine aria/silicio con diversi angoli di incidenza. L'entità della rifrazione (curvatura) dipende dalle proprietà dei due materiali ed in particolare dal loro indice di rifrazione. Per angoli di incidenza che superano un certo valore critico la luce rimane intrappolata nel silicio senza fuggire nell'aria, ottenendo la così detta riflessione totale. Un segnale luminoso può così propagarsi per chilometri all'interno della fibra senza subire perdite significative[2].

1.2 Confronto con altri mezzi trasmissivi

Come già accennato esistono parecchi vantaggi che privilegiano l'uso delle fibre rispetto ai cavi in rame nelle telecomunicazioni. In particolare l'uso della tecnologia ottica garantisce una bassa attenuazione, il che rende possibile la

trasmissione su lunga distanza senza ripetitori. Inoltre questa mette a disposizione una banda molto ampia in grado di garantire una grande capacità di trasporto oltre ad avere una pressochè totale immunità da interferenze elettromagnetiche, inclusi gli impulsi elettromagnetici nucleari (ma possono essere danneggiate da radiazioni alfa e beta). L'alta resistenza elettrica rende possibile usare le fibre vicino ad equipaggiamenti ad alto potenziale, o tra siti a potenziale diverso anche grazie al peso e ingombro modesto. Un cavo di fibra ottica, in quanto contiene più fibre ottiche, è infatti solitamente molto più piccolo e leggero di un filo o cavo coassiale con simili capacità di canale. È più facile da maneggiare e da installare ed è ideale per le comunicazioni sicure in quanto è molto difficile da intercettare e altrettanto facile da monitorare. Idealmente, le fibre ottiche sono un mezzo di trasmissione perfetto, infatti oltre a non risentire in nessun modo di disturbi elettromagnetici o di diafonia, se strutturate adeguatamente per garantire la riflessione totale del segnale d'ingresso, permettono teoricamente di trasferire completamente la potenza in ingresso nell'uscita e lavorando con fenomeni fisici ad elevatissima frequenza (le onde luminose) sarebbero possibili velocità di trasmissione molto elevate. In pratica, però, intervengono dei fattori fisici che limitano la banda di trasmissione possibile e causano delle perdite di potenza lungo la fibra[2].

1.3 La comunicazione dei dati su fibra ottica

Un sistema di comunicazione ottico è formato da tre componenti fondamentali: una sorgente luminosa (un Light Emitting Diode LED o un *laser* a semiconduttore), un mezzo di trasmissione (la fibra ottica) e un rilevatore (ad esempio un fotodiodo). Per convenzione si è stabilito che un impulso di luce corrisponda al valore 1, mentre l'assenza di luce indichi il valore 0. La sorgente di luce accetta in ingresso un qualsiasi segnale elettrico (binario), lo converte in una serie di impulsi luminosi che si propagano attraverso il mezzo trasmissivo. Grazie alla sue proprietà di riflessione totale, l'informazione sotto forma di luce viaggia senza dispersione all'interno della fibra. All'altra estremità del mezzo trasmissivo vi è un rilevatore che, quando è colpito dalla luce, genera un impulso elettrico riconvertendo così l'informazione dal dominio ottico a quello elettromagnetico. Nella figura precedente era rappresentato un solo raggio intrappolato nel mezzo di trasmissione, ma poichè tutti i raggi di luce che colpiscono l'interfaccia con un angolo maggiore di quello critico (per la riflessione totale) ee entro un cono di accettazione (per l'ingresso nella fibra) sono tutti riflessi internamente, una fibra può contenere molti raggi che rimbalzano ad angoli diversi. In questo caso si dice che ogni raggio ha una

modalità diversa.

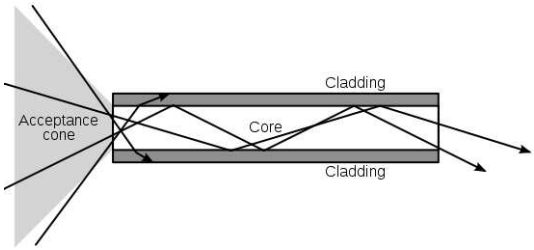

Figura 1.4: Principio di funzionamento di una fibra ottica multimodale [2]

All'interno di una fibra ottica quindi il segnale può propagarsi in modo rettilineo oppure essere riflesso un numero molto elevato di volte. Distinguiamo quindi due tipologie base di fibre:

- **Fibre ottiche monomodali**: se il diametro della fibra è ridotto a poche lunghezze d'onda al loro interno viaggia un solo raggio luminoso e la fibra si comporta come una guida d'onda ovvero la luce può propagarsi solo in linea retta detta anche modo di ordine zero.
- **Fibre ottiche multimodali**: al loro interno viaggiano più raggi luminosi a differenti angoli di incidenza, quindi consentono la propagazione di più modi (da qui il loro nome).

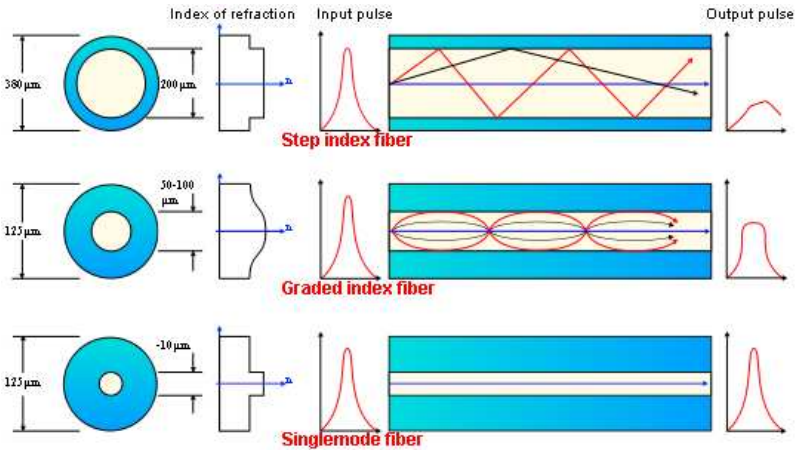

Figura 1.5: Differenza tra fibre ottiche multimodali (Step Index e Graded Index) e monomodali (o Single Mode) [2]

Le fibre multimodali permettono l'uso di dispositivi più economici, ma subiscono il fenomeno della dispersione intermodale, per cui i diversi modi si propagano a velocità leggermente diverse, e questo limita la distanza massima a

cui il segnale può essere ricevuto correttamente. Le fibre monomodali di contro hanno un prezzo molto più elevato rispetto alle multimodali, ma riescono a coprire distanze e a raggiungere velocità nettamente superiori[2].

1.3.1 Finestre di trasmissione

Nelle comunicazioni ottiche, lo spettro trasmissivo è descritto in termini di lunghezza d'onda (*wavelength*) invece che di frequenza. Combinando i diversi fenomeni di attenuazione, rifrazione, dispersione, vi sono delle finestre particolarmente adatte all'uso della trasmissione delle informazioni, con prestazioni e costi crescenti. Nel corso degli anni infatti, grazie al raffinamento del processo produttivo si è arrivati alla costruzione di fibre ottiche in grado di garantire buone prestazioni in termini di attenuazione e ampiezza trasmissiva. Nelle fibre prodotte negli anni '70, come visibile in figura, si avevano caratteristiche poco performanti che lasciavano spazio alla definizione di una sola finestra trasmissiva. Già negli anni '80 si è raggiunto un notevole miglioramento perfezionato nel decennio successivo con l'introduzione di ben tre finestre di trasmissione[2].

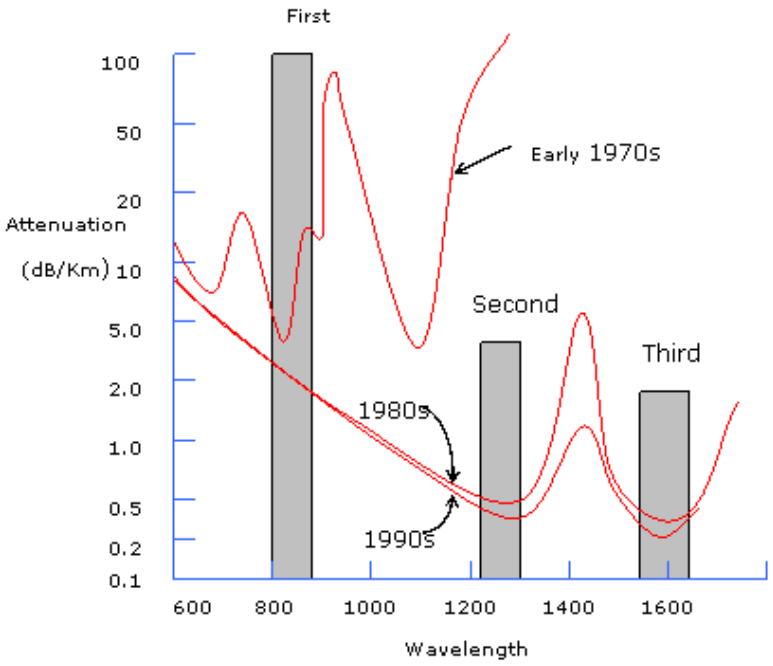

Figura 1.6: Relazione tra anni di produzione delle fibre ottiche e relative caratteristiche[21].

L'attenuazione della luce nella fibra è espressa solitamente in dB per chilometro lineare e si ricava dalla formula:

$$attenuazione = 10log_{10}\frac{energia\_trasmessa}{energia\_ricevuta}$$

Come accennato nelle moderne fibre ottiche sono disponibili dalle 3 alle 4 finestre di trasmissione poste dove le caratteristiche della fibra con una luce ad una determinata lunghezza d'onda garantiscono la minima attenuazione per chilometro possibile[2].

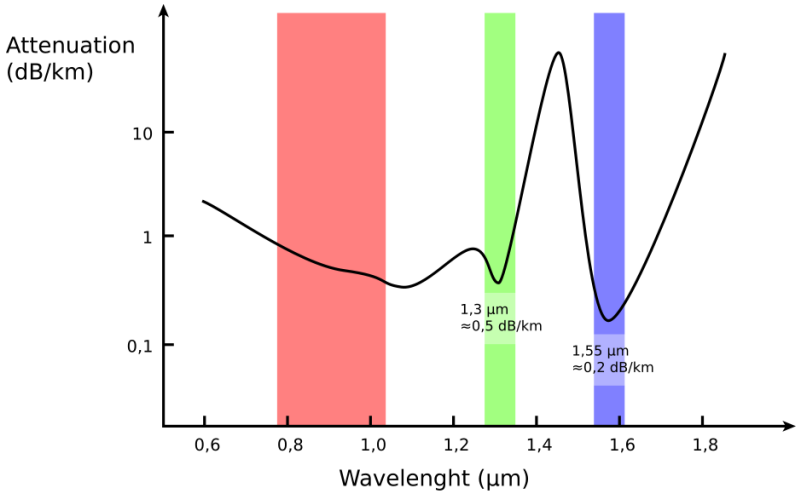

Figura 1.7: Finestre di trasmissione e lunghezze d'onda [2]

Queste finestre, come è possibile vedere nella figura, sono[2]:

- **Prima Finestra (rossa)**: 850 nm (nel campo del visibile), usata soprattutto con economici laser a diodo con luce multimodale. Permette di realizzare collegamenti di 275 m su fibre 62.5/125 e di 550 m su fibre 50/125.
- **Seconda Finestra (verde)**: 1310 nm, usata con laser multimodali o monomodali. Permette di realizzare collegamenti di 5 o 10 km su fibre monomodali.
- **Terza finestra (blu)**: 1550 nm, usata con laser monomodali. Questa finestra permette di realizzare le distanze maggiori, compresi collegamenti di 100 km con apparati relativamente economici. Sfruttando questa lunghezza d'onda, una buona fibra monomodale raggiunge una attenuazione dell'ordine degli 0,2-0,25 dB/km.

Tutte e tre le finestre o bande hanno ampiezze comprese tra i 25.000 e i 30.000 GHz, la seconda e la terza finestra hanno buone proprietà di attenuazione (meno del 5% di dispersione per chilometro) mentre la prima finestra ha

una attenuazione più alta ma, a questa lunghezza d'onda, è possibile costruire i laser e l'elettronica di supporto con l'uso dello stesso materiale per il drogaggio (arseniuro di gallio) risparmiando sui costi e semplificando il processo produttivo[2].

1.3.2 Tecniche di multiplazione

Nelle telecomunicazioni la multiplazione (in inglese *multiplexing*) è un meccanismo per cui la capacità disponibile di un collegamento viene condivisa tra diversi canali trasmissivi. Questo avviene combinando più segnali analogici o flussi di dati digitali in un solo segnale trasmesso su un singolo collegamento fisico, al fine di risparmiare nella comunicazione dei dati, e in particolare, di ridurre il numero di linee di segnale e il numero di componenti. La capacità del collegamento può essere suddivisa con diversi meccanismi[2]:

- a divisione di tempo o Time Division Multiplexing (TDM)
- a divisione di frequenza o Frequency Division Multiplexing (FDM)
- a divisione di lunghezza d'onda (o Wavelength Division Multiplexing (WDM) nelle comunicazioni ottiche)

La multiplazione a divisione di tempo ci permette di frazionare sull'asse dei tempi l'uso del canale in modo da assegnare ad ogni risorsa un quanto di tempo su cui può trasmettere, mentre la multiplazione a divisione di frequenza e di lunghezza d'onda, che è una forma particolare di multiplazione a divisione di frequenza in cui ogni canale trasmissivo viene inviato su una diversa lunghezza d'onda, e i canali possono essere combinati e separati restando nel dominio ottico, ovvero senza riconvertirli in segnali digitali[2].

1.3.3 Wavelength Division Multiplexing (WDM)

Questo tipo di multiplazione è utilizzato propriamente nei sistemi di comunicazione ottica. E' una variazione della multiplazione a divisione di frequenza dalla quale differisce sia perchè viene attuata a frequenze molto più alte (ovvero quelle proprie dei fasci di luce usati nelle fibre ottiche) sia perchè per la suddivisione ci si affida ad un sistema (ottico) completamente passivo (e quindi molto affidabile) basato su un reticolo di diffrazione[2][7]. Supponiamo di voler trasmettere 4 segnali ottici che trasportano ognuno energia ad una differente lunghezza d'onda su un unica fibra. Questi segnali, mediante l'azione di un combinatore ottico (che applica la multiplazione WDM), vengono uniti in un

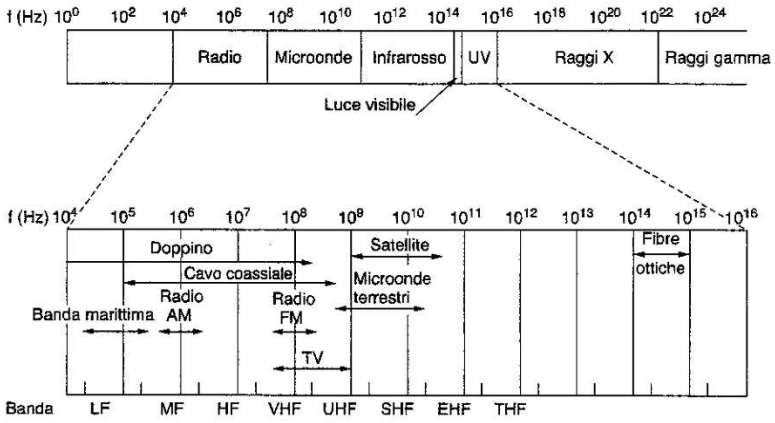

Figura 1.8: Lo spettro elettromagnetico, notare la posizione delle fibre ottiche[2]

unico spettro in modo da viaggiare su di un unica fibra ottica, verso una destinzione lontana anche migliaia di chilometri. Al suo arrivo la fibra è suddivisa in tante copie quante sono i segnali mediante uno *splitter*, poi viene filtrata da un filtro che permette solo ad una determinata banda di passare ottenendo così i segnali originali. Per modulare diversi canali su una stessa fibra ottica si usano diverse portanti di differenti lunghezze d'onda, una per ogni canale, e per la singola portante si usa la modulazione di intensità. In questo modo è possibile sfruttare la grande banda ottica disponibile. In gergo, le lunghezze d'onda vengono anche chiamate colori e la trasmissione WDM viene detta colorata, anche se in realtà le lunghezze d'onda usate non sono nel campo del visibile. Un sistema WDM usa un *multiplexer* in trasmissione per inviare più segnali insieme, e un *demultiplexer* in ricezione per separarli. I dispositivi di filtraggio ottico usati nei modulatori-demodulatori sono di solito degli interferometri di Fabry-Perot a stato solido e singola frequenza, nella forma di vetro ottico ricoperto da film sottile. I sistemi moderni possono gestire fino a 160 segnali e possono quindi moltiplicare la banda di una fibra a 10 Gbit/s fino a un limite teorico di oltre 1.6 Tbit/s su una singola coppia di fibre.

I sistemi WDM sono apprezzati dalle società telefoniche perché consentono di aumentare la banda disponibile in una rete senza dover stendere altra fibra ottica. Usando il WDM e gli amplificatori ottici, è possibile aggiornare progressivamente la tecnologia degli apparati di rete senza essere costretti a rifare totalmente la rete *backbone*. La capacità di banda di un certo collegamento può essere aumentata semplicemente aggiornando i multiplatori e demultiplatori a ciascun capo del collegamento. Essendo le reti ottiche basate su WDM in contatto ai loro estremi con reti normali basate su segnali elettromagneti-

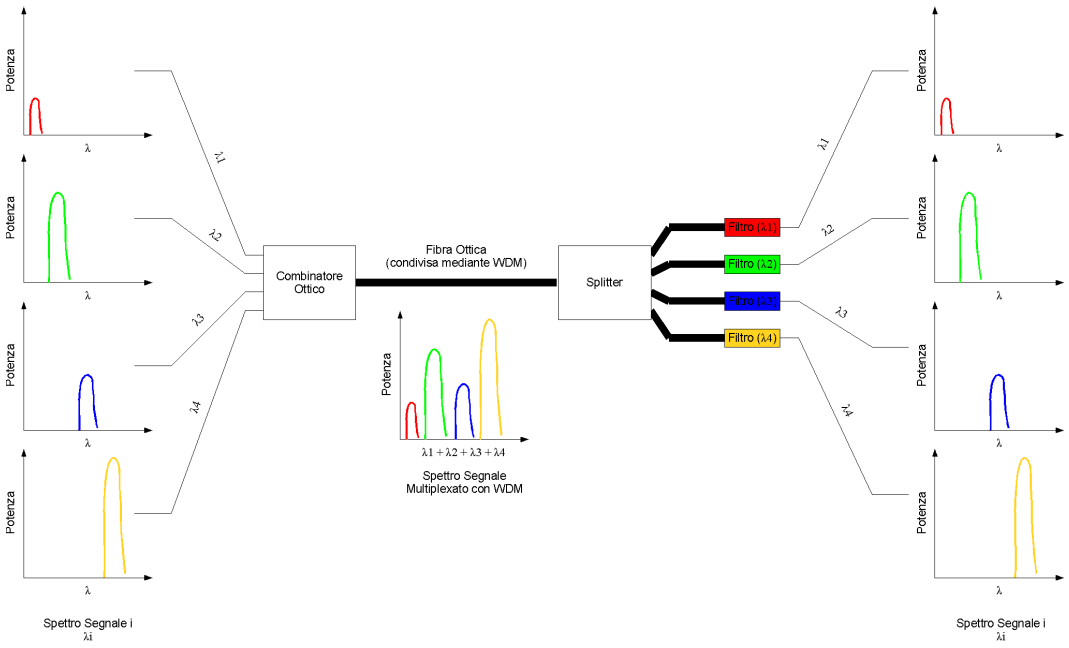

Figura 1.9: Tecnica di multiplazione WDM su fibra ottica.

ci, si renderà necessario ad un tal punto convertire il contenuto informativo nel dominio lettromagnetico. Questo è spesso realizzato compiendo una serie di conversioni Optical-Electrical-Optical (OEO) alle estremità della rete di trasporto, permettendo così l'interoperabilità con gli esistenti apparati con interfacce ottiche. L'energia su una singola fibra WDM è generalmente ampia pochi GHz in quanto non è ancor oggi possibile effettuare una conversione OEO in tempi più rapidi.

I sistemi WDM si possono suddividere, in base alla separazione tra le diverse lunghezze d'onda usate, in:

- **Conventional WDM o semplicemente WDM**: forniscono fino a 16 canali nella terza finestra di trasmissione (la banda C) delle fibre in silicio, intorno alla lunghezza d'onda di 1550 nm, con una separazione tra i canali di 100 Ghz.

- **Dense WDM o DWDM**: usa la stessa finestra di trasmissione ma con minore separazione tra i canali, arrivando a 31 canali a intervalli di 50 GHz; sistemi a 62 canali e intervalli di 25 GHz sono a volte chiamati ultra densi. Nuove possibilità di amplificazione (amplificazione Raman) consentono l'utilizzo anche delle lunghezze d'onda nella banda L, tra i 1570 nm e i 1610 nm, circa raddoppiando il numero di canali.

- **Coarse WDM o CWDM**: letteralmente a 'grana grossa', la separazione tra le lunghezze d'onda usate è maggiore che nel convenzionale e

nel DWDM, in modo da poter utilizzare componenti ottici meno sofisticati e quindi meno costosi. Per continuare a fornire 16 canali su una sola fibra, il CWDM usa interamente la banda di frequenze compresa tra la seconda e la terza finestra di trasmissione (1310/1550 nm rispettivamente) in cui, oltre alle due finestre (la finestra a minima dispersione e quella a minima attenuazione) è compresa anche l'area critica dove può aversi lo *scattering* (dispersione).

La tecnica WDM è semplice e il suo principio è noto da tempo, ma ha avuto applicazione pratica solo in tempi recenti. Il problema principale era la mancanza di amplificatori adatti. Da quando sono in uso le fibre ottiche, l'unico modo per superare lunghe distanze era la rigenerazione del segnale attraverso i rigeneratori optoelettronici. In un rigeneratore optoelettronico gli impulsi indeboliti vengono trasformati da un rivelatore fotoelettrico e, debitamente amplificati, modulano un trasmettitore laser. Il problema è che il rivelatore fotoelettrico non distingue una lunghezza d'onda da un'altra. La nuova invenzione che ha permesso il superamento di questa difficoltà è la tecnica che permette l'amplificazione della quantità di luce del segnale, senza bisogno di trasformarlo in segnale elettrico. Questi congegni, detti Erbium-Doped Fiber Amplifiers (EDFA) o amplificatori a fibra drogata con erbio, vennero sviluppati verso la fine degli anni '80 ed hanno reso possibile la rivoluzione basata su questa tecnica. La multiplazione su lunghezza d'onda è emersa al momento giusto, quando i vecchi cavi in fibra cominciavano ad essere saturi. L'esigenza di risparmiare tempo e denaro ha causato una rapida diffusione della tecnica WDM nell'industria delle telecomunicazioni. Ha evitato la spesa connessa con la posa di nuovi cavi semplicemente pompando altre lunghezze d'onda nelle fibre esistenti. Alla metà degli anni '90, le aziende hanno iniziato ad usare sistemi che trasmettono su 4 lunghezze d'onda e poco dopo sono salite ad 8 e poi a 16 (1996). Nel 1997 si è saliti a 32 e 40 bande, larghe appena 0.8 nm ciascuna. Nel 1998 si è giunti ad 80 canali. Visto che la domanda di larghezza di banda non sembra voler rallentare, si sta pensando al modo di stipare un numero maggiore di lunghezze d'onda in ogni fibra impiegando ad esempio tre amplificatori all'erbio ottimizzati per bande separate fra 1525 e 1605 nm[2][7].

1.4 Principi di commutazione ottica

Come accennato nell'introduzione, i requisiti futuri nelle specifiche di costruzione delle reti di telecomunicazioni sono molteplici e differenti, ma molti convergono con la richiesta di una capacità maggiore in termini di banda, flessibilità, robustezza, riduzione dei costi e di fornitura di energia, come di equipaggia-

mento e manutenzione. Tutto questo ha orientato il mondo delle telecomunicazioni verso la comunicazione basata su *backbone* a fibre ottiche, dato che questo mezzo trasmissivo raccoglieva ampiamente la maggior parte di questi requisiti. Scelta la tecnologia di base occorre stabilire come le informazioni permeranno la rete e come risolvere il percorso dal mittente al destinatario. La commutazione nelle telecomunicazioni riguarda i concetti generali sui quali è basato il funzionamento logico dei nodi di rete, ovvero è un'operazione all'interno di un nodo che tratta l'informazione da trasmettere, affinché sia indirizzata verso la destinazione desiderata. Una commutazione è attuata per mezzo delle funzioni d'instradamento o *routing* (decisionale) e di attraversamento o *forwarding* (attuativa). Il *framework* su cui si basano oggi le trasmissioni dati su fibra ottica prevedono una comunicazione punto-a-punto e funzioni di inoltro di tipo *packet switching* a controllo (ancora) elettronico, questo per una serie di problematiche legate alla difficoltà di implementare algoritmi di routing rimanendo nel dominio ottico. Infatti al giorno d'oggi nella tecnologia ottica sono principalmente due gli ostacoli sui quali la ricerca sta concentrando i propri sforzi:

- Non è possibile avere delle memorie sulle linee ottiche.
- La commutazione ottica è dispendiosa e complessa.

Vedremo di seguito come risolvere questi problemi e realizzare una struttura per l'uso della fibra ottica nelle comunicazioni a pacchetto.

1.4.1 Paradigmi di commutazione

Come abbiamo accennato la commutazione è una potenzialità di una rete di costruire, mantenere e abbattere un collegamento tra i nodi che la compongono. Esistono due grandi famiglie che si differenziano per le modalità in cui la commutazione avviene:

- Commutazione di circuito o multiplazione deterministica
- Commutazione di pacchetto o multiplazione statistica

Mentre in una rete a commutazione di circuito la capacità del canale trasmissivo è interamente dedicata ad una specifica comunicazione, la commutazione di pacchetto si rivela molto più efficiente nonostante la maggior quantità di dati inviata, in quanto i canali fisici sono utilizzati solo per il tempo strettamente necessario. Inoltre, poiché ogni pacchetto porta con sé la sua identificazione, una rete può trasportare nello stesso tempo pacchetti provenienti da sorgenti differenti. La commutazione di pacchetto permette quindi

a più utenti di inviare informazioni attraverso la rete in modo efficiente e simultaneo, risparmiando tempo e costi mediante la condivisione di uno stesso canale trasmissivo (cavo elettrico, etere, fibra ottica, ...). Storicamente la commutazione di pacchetto poneva qualche problema nel caso fosse necessaria una disponibilità garantita di banda o nelle trasmissioni *real time*: si pensi a una trasmissione video, dove le immagini arrivano con un flusso costante. Al giorno d'oggi è però possibile aggiungere una priorità ai pacchetti per garantire che un numero sufficiente di essi venga inviato, a scapito di altri pacchetti che non abbiano un'urgenza specifica, ad esempio, un file da trasferire[9].

In base a queste due grandi famiglie si sono definiti tre diversi paradigmi di commutazione ottica

- Optical Circuit Switching (OCS)
- Optical Packet Switching (OPS)
- Optical Burst Switching (OBS)

Optical Circuit Switching (OCS)

Fa propria la commutazione di circuito in ambito ottico cercando di costruire dei percorsi tra il mittente ed il ricevente mediante delle procedure, che viaggiano su un canale di segnalazione fuori banda per la mediazione, la creazione e l'abbattimento della connessione.

Optical Packet Switching (OPS)

Trasposizione della commutazione di pacchetto in ambito ottico, in cui viaggiano pacchetti e informazioni di instradamento sulla stessa banda (non esistono canali di segnalazione o controllo.

Optical Burst Switching (OBS)

Questo tipo di commutazione si trova tra la commutazione di circuito ottico e la commutazione di pacchetto ottico. Opera a livello di sub-lunghezza d'onda, il traffico in ingresso dai client ai margini della rete viene aggregato alla penetrazione della rete in base a un parametro particolare, comunemente per destinazione, tipo di servizio (TOS byte) la classe di servizio e qualità del servizio (ad esempio, secondo Diffserv). Pertanto, al router edge OBS, code diverse rappresentano le varie destinazioni o la classe di servizi.

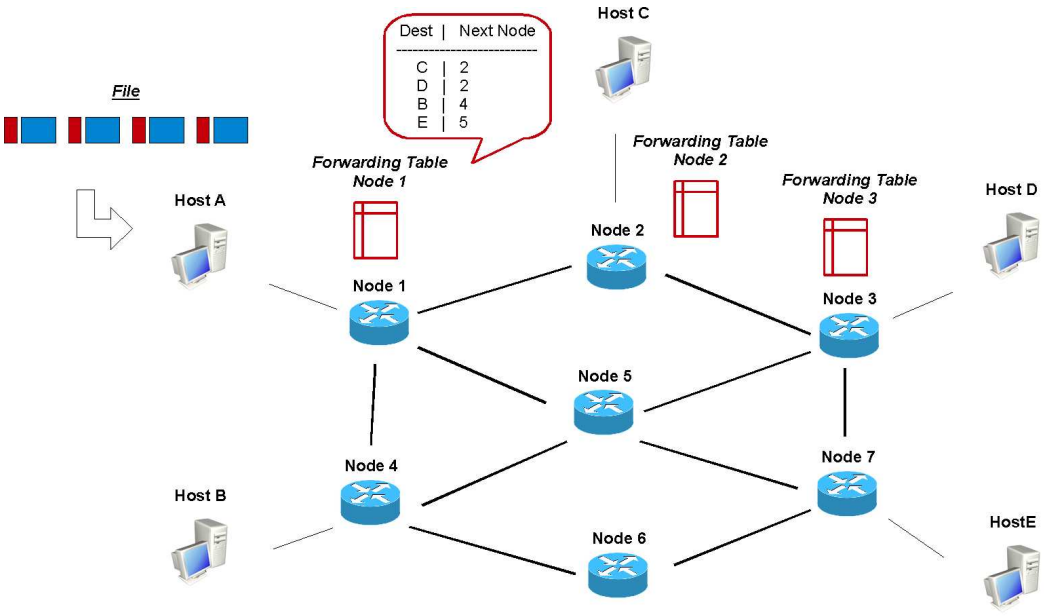

Figura 1.10: Esempio di scenario tipico della commutazione a pacchetti.

1.5 Optical Packet Switching (OPS)

Vedremo ora in dettaglio il paradigma OPS analizzando in primis uno scenario tipico di utilizzo, mostrandone i punti di forza che ne hanno favorito lo sviluppo e la diffusione.

1.5.1 Packet Switching Scenario

Supponiamo di voler trasmettere un file da un *host* ad un altro attraverso una rete (ad esempio dall'*host* A all'*host* D). Il file viene suddiviso in pacchetti (segmenti blu) e vengono inseriti degli *header* ad ognuno di essi (segmenti rossi) contenenti le informazioni sull'*host* destinazione da raggiungere. I pacchetti viaggiano nella rete e in ogni nodo intermediario che incontrano vi è una tabella di inoltro (*forwarding table*) che contiene l'informazione riguardante il percorso che ogni pacchetto deve seguire per una determinata destinazione. Come abbiamo evidenziato, ricorriamo al *packet switching* perchè ci permette una allocazione dinamica della banda (i collegamenti saranno occupati solo quando necessario ed esistono delle alternative routes quando si verificano congestioni) inoltre i pacchetti da differenti sorgenti possono coesistere sulla stessa rete permette di far lavorare terminali a differenti *bit rates*[18].

Nelle reti MAN quindi generalmente i link tra i nodi sono realizzati mediante l'uso di fibre ottiche e i pacchetti che viaggiano al suo interno sono dei pacchetti ottici. Per utilizzare la commutazione di pacchetto è dunque

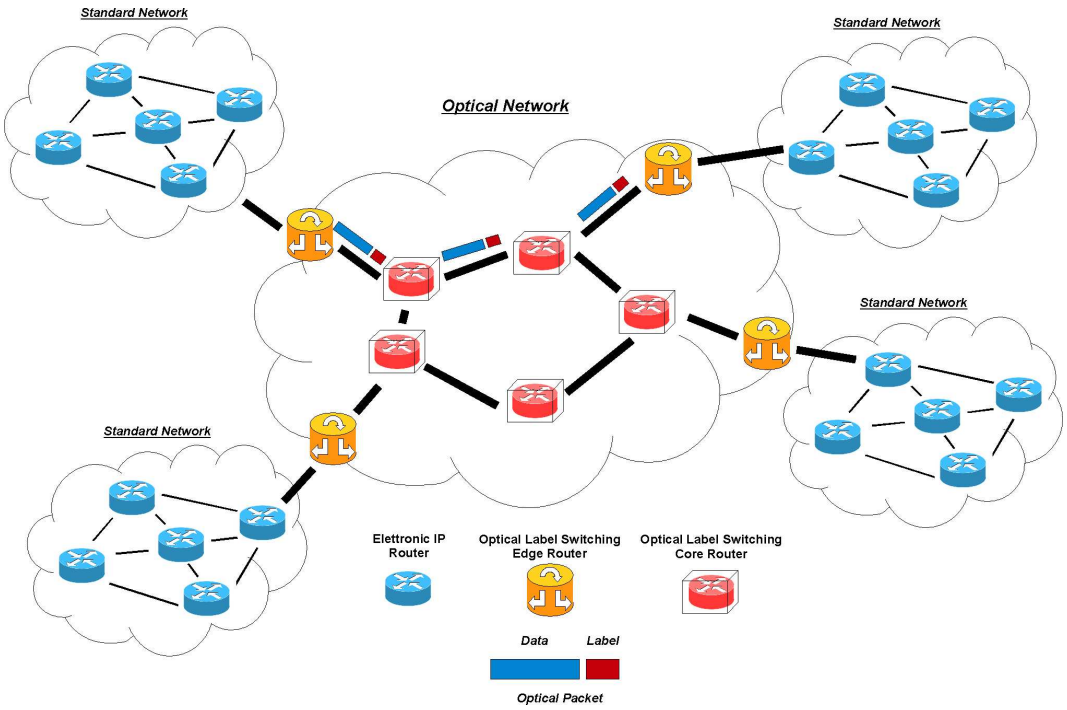

Figura 1.11: Esempio di rete Optical Packet Switching (OPS)

necessaria una conversione di tipo Optical-Electrical-Optical (OEO) alle interfacce per operare lo switching dei pacchetti. Questa operazione di conversione dal segnale ottico a quello elettronico genera un elevato overhead in quanto è una operazione lenta per le tempistiche della rete ottica (quindi per essa penalizzante) limitata nel numero di pacchetti processabili contemporaneamnete costosa e complessa oltre ad introdurre un difetto proprio delle comunicazioni elettriche come la diafonia (cross-talk) ovvero il rumore o interferenza elettromagnetica che si può generare tra due cavi vicini di un circuito o apparato elettronico. Queste limitazioni nella conversione OEO degradano le performance della fibra diminuendone la banda trasmissiva[18].

La soluzione è individuata nella tecnica dell Optical label Switching (OLS) che permette di instradare i pacchetti ottici all'interno della rete di fibra rimanendo nel dominio ottico senza convertire il segnale nel campo elettromagnetico. Si posiziona un etichetta (*label*) sulla frequenza portante del pacchetto, così facendo le informazioni di instradamento possono essere estratte convertendo il solo *header* e lasciando così il *payload* intatto nel dominio ottico[18].

1.5.2 Optical Label Switching OLS

Questa tecnica viene utilizzata per costruire su una rete a pacchetto diverse reti virtuali indipendenti e reciprocamente impermeabili. A ciascun pacchetto

viene aggiunta una etichetta che lo identifica come appartenente ad una particolare rete virtuale, e potrà essere consegnato solo se il destinatario appartiene alla stessa rete virtuale. Tra gli esempi, le VLAN su Ethernet (802.11q) e Multi Protocol Label Switching (MPLS), che è un meccanismo che crea reti private virtuali su tecnologie eterogenee, tipicamente utilizzato dai provider[18].

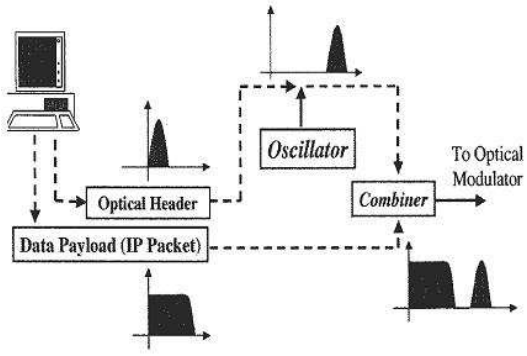

Figura 1.12: Modulazione della etichetta nel pacchetto ottico [18]

Funzionamento

Nell'OLS si posiziona dunque una label (con le informazioni di switching per i nodi ottici) ad una sottofrequenza della portante come mostrato in 1.12. Così facendo le informazioni possono essere estratte convertendo (elettronicamente) il solo *header* lasciando il *payload* nel dominio ottico.

1.5.3 Il pacchetto ottico

Quando trasmettiamo i pacchetti nel dominio ottico, dobbiamo definire anche quale forma hanno i dati inviati. Una proposta che tende a modellare una versione ottica della cella ATM prevede che il pacchetto sia composto da quattro componenti fondamentali, un *header* con le informazioni di *label switching* per l'instradamento ottico, il *payload* e due bande di guardia (*guard bands*) che facilitano nella individuazione delle due sezioni durante l'analisi del segnale[18] come mostrato in 1.13.

Come si evince in figura, le due sezioni importanti sono l'*header* e il *payload*. Mentre quest'ultimo rappresenta un incapsulamento delle informazioni che viaggiano ad un livello superiore a quello di trasporto su fibra ottica, tutte le informazioni che riguardano l'instradamento dei pacchetti in OPS sono contenute nell'header, che è composto da diversi campi[19]:

- **Sync**: Contiene dei bit di sincronizzazione.

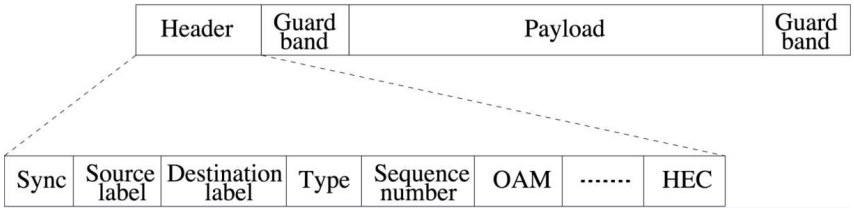

Figura 1.13: Struttura del pacchetto ottico secondo proposta OPATM/KEOPS

- **Source Label**: Etichetta del nodo sorgente del pacchetto.
- **Destination Label**: Etichetta del nodo destinazione del pacchetto.
- **Type**: Tipo e priorità del payload trasportato.
- **Sequence Number**: Numero sequenziale del pacchetto per riordinare i pacchetti all'arrivo e garantire una consegna in ordine.
- **OAM**: Operation, Administration, Maintenance.
- **HEC**: Head Error Correction.

1.5.4 Categorie di reti OPS

Esistono fondamentalmente due grandi famiglie di reti OPS, le prime dette *Slotted* OPS *Networks* suddividono il tempo in slot di dimenione fissa al cui interno possono presentarsi o meno dei pacchetti, e reti di tipo *Unslotted* OPS *Networks* in cui abbiamo un approccio meno rigido in cui ogni pacchetto è considerato al suo istante di arrivo. Vediamo ora in dettaglio le due cateogrie[20].

Slotted OPS networks

Nelle reti OPS di tipo slotted i pacchetti che viaggiano all'interno sono di dimensione fissa e sono posizionati all'interno di intervalli di tempo prefissati detti time slots. Ogni time slot può contenere un singolo paccheto (composto come visto da header, payload e le bande di guardia addizionali). I nodi su questa tipologia di rete lavorano in maniera sincrona mantenendo i confini dei time slot allineati fra loro. Inoltre l'uso di switch sincroni in reti di tipo slotted ci permette di dover risolvere un minor numero di contese tra pacchetti (dato che questi sono di lunghezza fissa e vengono inoltrati insieme con i limiti degli slot allineati) oltre a poter portare pacchetti già nativamente di dimensione fissa (come ad esempio nelle celle ATM). Come contro questa tipologia richiede l'allineamento dei pacchetti e uno stadio di sincronizzazione[20].

Unslotted OPS networks

Nelle reti OPS di tipo unslotted i pacchetti che viaggiano all'interno possono essere di dimensione variabile, e il tempo non è suddiviso in slot. I nodi su questa tipologia di rete lavorano in maniera asincrona senza alcun requisito di allineamento. Inoltre l'uuso di switch di tipo asincrono in reti di tipo unslotted ci permette di evitare la segmentazione e il riassemblamento dei pacchetti sia in ingresso sia in uscita dai nodi, è in grado di trasportare pacchetti di dimensione variabile (ad es. su protocollo IP) ma presenta un numero maggiore di contese da risolvere[20].

1.5.5 Content Resolution

Uno dei principali problemi dell'approccio di tipo *Optical Packet Switching (OPS)* è la così detta *Contention Resolution* (o risoluzione di contesa) che si verifica quando due o più pacchetti competono per la stessa risorsa (nel caso ottico stessa fibra e lunghezza d'onda) allo stesso istante[20]. Questo problema può essere affrontato e risolto in diversi domini:

- Spazio
- Tempo
- Frequenza o più propriamente lunghezza d'onda
- ...

Mentre però nella commutazione di pacchetto su dominio elettromagnetico Electronic Packet Switching (EPS)) il problema della contesa è affontato operando nel dominio del tempo usando delle memorie RAM come code, in ambito ottico Optical Packet Switching (OPS)) questo approccio si scontra con un limite tecnologico tuttora irrisolto. Ad oggi infatti non si è riusciti ad implementare un componente che agisca come memoria ottica, in modo equivalente alle RAM elettroniche, e per ovviare a questa carenza si ricorre a delle linee di ritardo che permettano alla logica di instradamento di stabilire il percorso ritardando al proprio interno il flusso dati, fino a quando l'elaborazione e il percorso di uscita non siano stabiliti. Questi componenti prendono il nome di Fiber Delay Line (FDL). Inoltre le FDL offrono solo valori di ritardo discreti e in numero limitato e contribuiscono ulteriormente al degrado della qualità del segnale. D'altra parte però l'approccio OPS permette di affrontare la contesa nel dominio delle lunghezze d'onda (*wavelength domain*) sfruttando le proprietà della *wavelength conversion*. Mediante dei particolari dispositivi

detti Wavelength Converter (WC) è possibile, in caso di contesa sulla stessa lunghezza d'onda, far variare la lunghezza d'onda di appartenenza di un pacchetto risolvendo così la contesa. Purtroppo i WC sono componenti molto complessi e costosi e, come vedremo, il loro utilizzo è molto razionalizzato nelle architetture di commutazione alla base delle reti ottiche[20].

1.5.6 Riflessioni sull'uso di OPS

Le caratteristiche che fanno di OLS una tecnica vincente sono da ricercare nei suoi punti di forza. OLS permette l'uso di una singola tecnologia (quella ottica) tra le *end-stations* della rete, quando un pacchetto lascia un host ed entra in una rete ottica gestita da OLS esso vede solo un unica lunga fibra ottica che lo conduce all'uscita alla rete di destinazione. Questò è concesso grazie al disaccoppiamento tra *data payload* e *header informations* spesso trasmessi a differenti livelli di *bit rate* (per facilitare il processamento elettronico dell'header). Inoltre i pacchetti che vengono inoltrati attraverso uno stesso percorso (path) sperimentano lo stesso ritardo, e se un pacchetto è bloccato ad un determinato nodo può essere rediretto ad un altro percorso o buttato via (*dropped*). Inoltre OLS introduce delle informazioni per il timing consideration. Il contention control (controllo sulla contesa) è effettuato mediante la deviazione su un altra lunghezza d'onda. Ovvero quando dei pacchetti provenienti da molteplici utenti arrivano allo switch node allo stesso tempo, posso indirizzare un pacchettoa ad una lunghezza d'onda meno sovraccarica.

Per quanto riguarda l'impatto a livello tecnologico ed economico l'uso di OLS offre numerosi vantaggi. Ad esempio il suo uso permette di colmare il gap tra l'uso di IP nelle normali reti e la tecnica OLS delle dorsali ottiche nei punti di contatto fra queste, sostituisce l'esistente topologia ad anello delle MAN con quella a optical switching, fornirà la base per i servizi della prossima generazione e semplificherà i costi di gestione per i service provider (veloci e semplici da gestire)

Riassumendo abbiamo visto che la tecnica di electrical packet switching non è compatibile con la trasmissione su tecnologia ottica rendendo necessario l'uso della tecnica detta di Optical Label Packet Switching (OLPS) che ci permette di evitare le conversioni di tipo OEO (prestazioni) garantendo al tempo stesso la compatibilità con numerosi *layer protocols* (mediante OLS). Inoltre OLPS è compatibile con la multiplazione in tecnica WDM fornendo dei miglioramenti prestazionali e, sulla giusta scala, economico in rapporto ad un proporzionale incremento della complessità[8].

Capitolo 2

Architetture per router ottici

Bisogna stare attenti agli ingegneri. Cominciano con le macchine da cucire e finiscono con la bomba atomica.

Marcel Pagnol, scrittore e regista francese.
(1895 - 1974)

Esistono differenti approcci nella progettazione di architetture per reti ottiche. Tutto ruota intorno alle necessità di progetto e alle specifiche in termini di prestazioni, funzionalità e costi. Essendo la tecnologia ottica ancora in uno stadio precedente alla diffusione su larga scala il prezzo dei singoli componenti è ancora particolarmente elevato e questo spesso spinge alla progettazione di architetture che condividano il più possibile l'uso di questi dispositivi, in un ottica di contenimento dei costi. In un approccio invece più orientato alla fornitura di servizi con elevata Quality of Service (QoS), in cui è importante garantire la presenza di flussi dati di tipo *burst* o pacchetti, senza avere vincoli e limitazioni nell'uso di componenti ottici, allora affiorano architetture più complete ma più costose. E' proprio in questo caso dove risulterebbe utile una piattaforma per il collaudo e la valutazione dell'architettura proposta prima della sua realizzazione e quindi dell'acquisto dei componenti che la realizzano. Introdurremo ora una generica architettura di router per reti ottiche OPS, in modo da introdurre i principali componenti in gioco, le tecnologie con cui vengono implementati e i rispettivi ruoli per poi presentare una carrellata delle principali architetture note in letteratura.

2.1 Generica architettura di un router ottico

Prima di analizzare a fondo le caratteristiche e le diverse potenzialità delle varie architetture, introduciamo una schematizzazione della struttura di base di uno di questi nodi per reti ottiche in modo da presentare i componenti principali e i loro ruoli all'interno del meccanismo di interconnessione svolto dal nodo stesso.

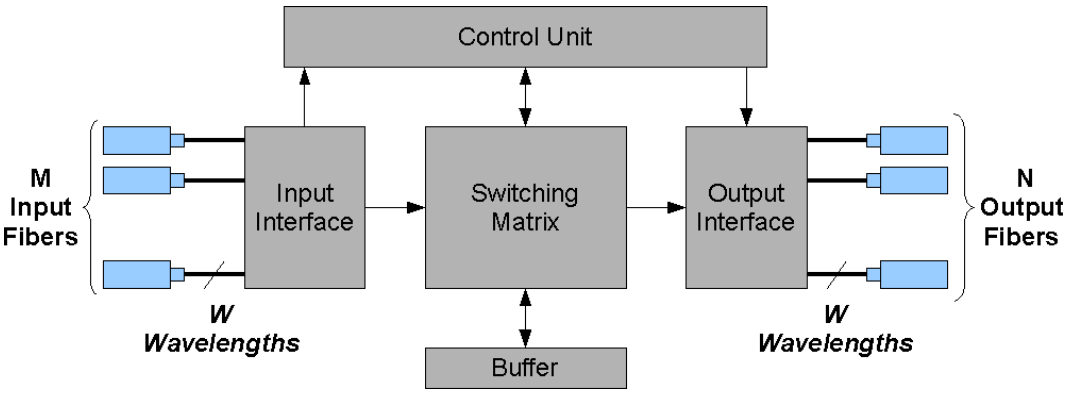

Figura 2.1: Una generica architettura per router ottici

Come visibile in figura 2.1, l'architettura di base si compone di diverse unità o blocchi. L'interfaccia di ingresso, quella di uscita, la matrice di commutazione, uno stadio di buffering e un unità di controllo. Nel seguito forniremo nozione dei principali ruoli di queste unità nel processamento dei pacchetti ottici all'interno del router al fine di chiarire meglio le interazioni che avvengono tra queste ed il percorso del traffico attraverso il nodo.

2.1.1 Input Interface

L'interfaccia di ingresso è il primo componente del router ottico che viene attraversato dal traffico in arrivo. Qui le informazioni vengono identificate, misurate e preparate al processamento nel nodo. In particolare possiamo identificare tre distinte fasi:

- Delineazione del pacchetto
- Sincronizzazione
- Pre-processamento dell'header

Delineazione del pacchetto

In questa fase l'interfaccia di ingresso identifica l'inizio e la fine del pacchetto, ed in particolare dell'header e del payload che lo compongono grazie all'aiuto

delle due *guard band* concepite per favorire questo scopo. Questa operazione separa quindi le informazioni di routing dal carico pagante operando la prima grande distinzione tra piano dati e piano di controllo. A livello di potenza, la delineazione dell'header e la sua successiva divisione dal payload non comporta una perdita significativa in termini di potenza globale del segnale. In questo lavoro quindi, come dettagliato in seguito non si terrà conto di questa variazione anche se in futuro, in una versione maggiormente dettagliata della corrente implementazione, questa modellazione può essere facilmente considerata.

Sincronizzazione (solo in switch sincroni)

Solo per quanto riguarda gli *switch* basati su un meccanismo di tipo *slotted* sincrono è prevista una fase di sincronizzazione ed allineamento dei pacchetti che arrivano da differenti lunghezze d'onda e differenti fibre di ingresso, considerati all'interno dello stesso timeslot.

Pre-processamento dell'header

Prima di lasciare l'interfaccia di ingresso il pacchetto è dunque separato dal payload come accennato, successivamente convertito dal dominio ottico a quello elettronico mediante un convertitore di tipo Optical-Electrical (OE), quindi decodificato e inoltrato all'unità di controllo per il processamento elettronico. Il payload invece, separato dall'header ma ancora nel dominio ottico viene posto in ingresso alla matrice di commutazione, o meglio nel buffer ad essa collegata, in attesa del processamento delle informazioni di routing che lo riguardano.

2.1.2 Control Unit

L'unità di controllo è il secondo componente in ordine di processamento all'interno del nodo, che si pone però su un piano diverso da quello dei dati finora coinvolto, e attiene più propriamente al piano detto, appunto, di controllo. Qui le informazioni contenute nell'header arrivano in forma elettronica, quindi sono più facili da elaborare grazie ai paradigmi, alle tecnologie e agli algoritmi di scheduling noti e disponibili da tempo in ambito elettronico ed informatico, a differenza delle neo tecnologie in ambito ottico. Queste informazioni, elaborate per risolvere le necessità del traffico in termini di qualità e destinazione da raggiungere, determinano quindi la configurazione della matrice di commutazione. Qui si delina dunque un punto di contatto tra il piano di controllo (decisionale) e il piano dati (attuatore). Si vedrà in seguito come alcune raccomandazioni di enti internazionali stiano definendo e cercando uno standard

per questa interazione. Una volta quindi processate le informazioni di routing e opportunamente configurata la matrice per il pacchetto in esame, il suo header viene aggiornato con le nuove informazioni che gli saranno necessarie all'uscita dal nodo per dialogare con il prossimo intermediario nel percorso tra il mittente e il destinatario, e spedito all'interfaccia di uscita.

2.1.3 Buffer

Questa unità in realtà, in base alle diverse architetture, può far parte o meno della matrice di commutazione, ed esservi posta in incipit o in uscita dalla stessa. La sua funzione però non cambia a prescindere dal suo posizionamento. All'interno di questa unità infatti il payload è in attesa che l'unità di controllo configuri opportunamente la matrice di commutazione nei termini dei dispositivi interessati, secondo le informazioni presenti nel suo header. Questa attesa, fissa o con una componente variabile, sempre in base alle varie architetture disponibili, è realizzata in ambito ottico mediante delle Fiber Delay Lines (FDL), componenti che ritardano il segnale ottico che le attraversa di un certo valore. Come accennato il ruolo che nel dominio elettronico è affidato a delle memorie RAM qui è svolto da questi componenti, se pur in maniera non proprio analoga.

2.1.4 Switching Matrix

La matrice di commutazione ottica è il componente più complesso e caratterizzante dell'intera architettura. Introdurremo qui solo le nozioni base della logica di funzionamento rimandando un dettagliato approfondimento sui componenti interni della struttura che ne delineano il funzionamento nel prossimo capitolo. La matrice di commutazione inoltra il payload attraverso i suoi dispositivi interni in accordo alle direttive ricevute dall'unità di controllo con lo scopo di convogliare il traffico in ingresso verso le sue uscite cercando ove possibile di risolvere eventuali contese mediante la conversione su lunghezze d'onda differenti.

2.1.5 Output Interfaces

L'interfaccia di uscita è l'ultimo componente del router ottico che viene attraversato dal traffico che attraversa il nodo. Qui le informazioni aggiornate provenienti da piano di controllo e quelle in uscita dalla matrice di commutazione vengono riunite, misurate e preparate a lasciare il nodo. In particolare possiamo identificare tre distinte fasi, mostrate di seguito.

Aggiornamento dell'header

Innanzitutto l'header aggiornato proveniente dall'unità di controllo viene riconvertito al dominio ottico mediante un convertitore di tipo Electrical-Optical (EO). Successivamente questo viene unito al payload proveniente dalla matrice di commutazione ricreando così il pacchetto ottico iniziale, aggiornato però nelle sue informazioni di instradamento ed inoltro. In questa operazione vengono sempre inserite le due *guard band* per distinguere le varie parti del segnale prima di spedirlo verso l'uscita.

Rigenerazione 3R

All'interno dell'interfaccia di uscita effettua spesso una rigenerazione del segnale ottico in termini di amplificazione di potenza, squadratura del segnale e temporizzazione. Questa operazione è detta rigenerazione 3R dalle iniziali in inglese delle tre operazioni *reamplification*, *reshaping* e *retiming*.

Sincronizzazione (solo in switch sincroni)

Solo per quanto riguarda gli switch basati su un meccanismo di tipo *slotted* sincrono è prevista una fase di sincronizzazione ed allineamento dei pacchetti che arrivano da differenti lunghezze d'onda per differenti fibre di uscita, considerati all'interno dello stesso timeslot.

2.2 La matrice di commutazione ottica

Come già accennato la matrice di commutazione ottica è il componente più complesso e che realmente differenzia le deiverse architetture. Forniremo ora una descrizione di base dei principali componenti che costituiscono questo complesso dispositivo al fine di chiarire le dinamiche e le principali differenze tra le varie architetture proposte nel prossimo capitolo.

2.2.1 Erbium-Doped Fiber Amplifiers (EDFA)

Questo dispositivo è l'amplificatore più distribuito per le soluzioni in fibra ottica, e la sua finestra di amplificazione coincide con la terza finestra di trasmissione del silicio sulla fibra ottica [2].

Figura 2.2: Simbolo grafico di un amplificatore EDFA

2.2.2 WDM Demultiplexer

Questo componente è in grado di ripartire il segnale ottico presente su una singola fibra ottica in ingresso, nelle sue W componenti pari alle lunghezze d'onda presenti.

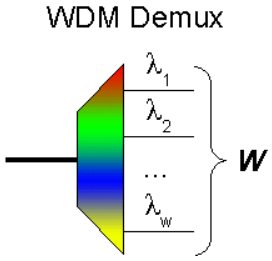

Figura 2.3: Simbolo grafico di un demultiplatore WDM

Grazie a questa demultiplazione si è in grado di manipolare le singole lunghezze d'onda, e quindi le informazioni presenti sulla stessa, senza preoccuparsi della contemporanea presenza di ulteriori segnali presenti ad altre frequenze o lunghezze d'onda.

2.2.3 Fiber Delay Line (FDL)

Un *buffer* ottico è un dispositivo che è in grado di immagazzinare temporaneamente la luce. Proprio come nel caso di un normale *buffer* standard, è un supporto di memorizzazione che consente la compensazione nella differenza dei tempi del verificarsi degli eventi. Più precisamente, un *buffer* ottico è in grado di memorizzare i dati trasmessi otticamente, senza la conversione al dominio elettrico. Inoltre le operazioni di processamento elettronico sono il vero collo di bottiglia dell'OPS, oltre al problema della generazione delle contese.

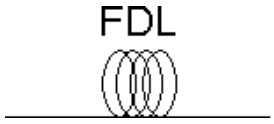

Figura 2.4: Simbolo grafico di una FDL

Ogni volta che due o più pacchetti di dati arrivano ad un nodo della rete, nello stesso istante e contendono per la stessa uscita, si verifica una situazione di contesa. La scelta più ovvia ma meno performante sarebbe quella di abbandonare tutti i pacchetti in eccesso, ma si possono applicare in genere tre soluzioni: il buffering, deflection routing o la conversione di lunghezza d'onda. Il buffering ottico utilizza delle linee di ritardo in fibra o Fiber Delay Line (FDL) per ritardare la luce, ed è considerato come il più efficace. Siccome la luce non può essere congelata, un buffer ottico è costituito da fibre ottiche, ed è generalmente molto più grande di un chip di memoria RAM di capacità comparabile. Una singola fibra può servire come un buffer. Tuttavia, una serie di più di solito è usato. Una possibilità, per esempio, è quello di scegliere un certo T per la fibra più piccola, e poi lasciare che il secondo, terzo, ... hanno lunghezze 2T, 3T, ... Un altro esempio tipico è quello di utilizzare un ciclo unico, in cui i dati circola un numero variabile di volte. [2]

2.2.4 Wavelength Converter (WC)

Un dispositivo in grado di convertire la lunghezza d'onda di appartenenza del sengale di ingresso ad un'altra lunghezza d'onda in uscita è detto convertitore di lunghezza d'onda o più propriamente *Wavelength Converter* o WC.

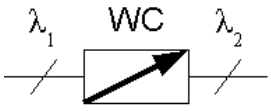

Figura 2.5: Simbolo grafico di un convertitore WC

Il classico convertitore di lunghezza d'onda è costituito da uno schermo di materiale luminescente, che assorbe le radiazioni luminose e le irradia in un lunghezza d'onda superiore. Esistono differenti tipologie di convertitori in base alla varietà.

Fixed-input Tunable-output Wavelength Converter (FTWC)

Convertitore che accetta solo un determinata lunghezza d'onda fissa in ingresso mentre permette di convertirla verso l'uscita su una qualsiasi altra lunghezza d'onda. Rappresenta il più economico dei dispostivi della sua categoria.

Tunable-input Tunable-output Wavelength Converter (TTWC)

Convertitore che accetta in ingresso una qualsiasi lunghezza d'onda e permette di convertirla verso l'uscita su una qualsiasi altra lunghezza d'onda. Più costoso

dei precedenti, permette però l'uso di un singolo dispositivo per un insieme di lunghezze d'onda al posto di W dispositivi di tipo FTWC.

2.2.5 Splitter

Grazie a questo dispositivo il segnale in ingresso proveniente da una singola fibra ottica può essere riprodotto, e quindi duplicato, verso un numero N di copie in uscita.

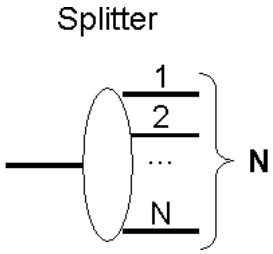

Figura 2.6: Simbolo grafico di uno splitter

A livello di caratterizzazione di potenza è ovvio che la duplicazione del segnale ne comporta anche una sua attenuazione proporzionale al numero di copie in uscita dal dispositivo. L'entità di questa attenuazione è pari al valore espresso nel modello dalla formula di attenuazione ideale:

$$attenuazione = 10log_{10}N$$

2.2.6 Switches

I veri dispositivi alla base del meccanismo di commutazione all'interno della matrice sono i cosidetti switch o interuttori. Questi componenti sono in grado di inoltrare o bloccare in base al loro stato (chiuso/aperto o ON/OFF) il passaggio dell'informazione attraverso di loro. Esistono dunque tecnologie e meccanismi differenti per realizzare tali dispositivi, ognuno dei quali presenta differenti caratteristiche e prestazioni in termini di velocità di commutazione dello stato, consumi e costi di realizzazione. Accenniamo ora due delle tecniche base per la costruzione degli switch all'interno della matrice di commutazione.

Semiconductor Optical Amplifier (SOA)

Gli amplificatori ottici a semiconduttore sono dei dispositivi che utilizzano un semiconduttore come mezzo per fornire il guadagno di potenza del segnale. Amplificatori di questo tipo vengono utilizzati nei sistemi di telecomunicazione

sia come puri e semplici amplificatori, sia come dispositivi interuttori. Tipicamente operano a lunghezze d'onda del segnale tra 0,85 micron e 1,6 micron e sono in grado di generare un guadagno fino a 30 dB[7].

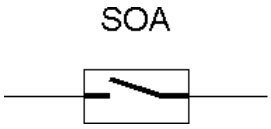

Figura 2.7: Simbolo grafico di un SOA

L'amplificatore ottico a semiconduttore è tipicamente di piccole dimensioni, e questo favorisce il processo di integrazione dei componenti nella matrice e inoltre è comandato (pompato) elettricamente. A certi livelli di scala, il SOA risulta meno costoso di un EDFA e può essere integrato con laser a semiconduttore, modulatori, e altri dispositivi, tuttavia, le prestazioni non sono ancora paragonabili a quelle di un EDFA. Infatti un SOA genera un rumore più alto e fornisce un minore guadagno. Ciò deriva dal tempo di vita dello stato, che dura approssimativamente qualche nanosecondo o poco più, ed è proprio per questo che il guadagno reagisce rapidamente alle variazioni di potenza del segnale che possono causare cambiamenti di fase e dar luogo ad alterazione dei segnali.

Micro-Electro-Mechanical Systems (MEMS)

Con questo termine si racchiude tutta la tecnologia dell'infinitamente piccolo, che porta a scala micrometrica i sistemi elettromeccanici. I MEMS sono anche denominati micromacchine o Micro Systems Technology (MST) e sono costituiti da componenti da 1 a 100 micrometri in termini di dimensioni (vale a dire 0,001-0,1 mm) e i dispositivi in generale variano nel formato da 20 micrometri (20 milionesimi di metro), a un millimetro[2]. In ambito ottico questi dispositivi vengono utilizzati per creare array di micro specchi riflettenti usati poi come dispositivi interuttori.

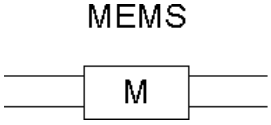

Figura 2.8: Simbolo grafico di un MEMS ad uso ottico

2.2.7 WDM Multiplexer

Questo componente è in grado di ricomporre a partire dalle sue W componenti pari alle lunghezze d'onda presenti, il segnale ottico da presentare in uscita

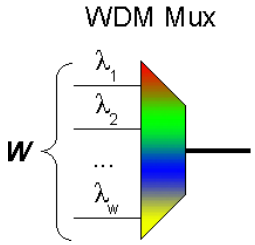

Figura 2.9: Simbolo grafico di un multiplatore WDM

Grazie a questa multiplazione si è in grado di rigenerare il segnale ottico da trasmettere in uscita con tutte le proprietà e le potenzialità che la tecnica di multiplazione WDM garantisce per la trasmissione dei dati a lunga distanza.

2.2.8 Combiner

Grazie a questo dispositivo il segnale in ingresso disponibile su M copie può essere ricombinato per tornare, in uscita, sotto forma di un singolo segnale ottico.

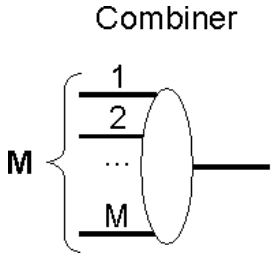

Figura 2.10: Simbolo grafico di un combiner

Anche qui a livello di caratterizzazione di potenza la riduzione dei sengali ne comporta anche una sua attenuazione proporzionale al numero di copie presenti in ingresso al dispositivo. L'entità di questa attenuazione è pari al valore espresso nel modello dalla formula di attenuazione ideale:

$$attenuazione = 10log_{10}M$$

2.3 Shared Architectures

Come abbiamo visto in ambito ottico il problema della contesa (*content resolution*) si affronta nel dominio delle lunghezze d'onda sfruttando le proprietà della *wavelength conversion*. Dato però che i WC sono componenti complessi e costosi, molte delle architetture proposte dalla comunità di ricerca considerano che i WC siano condivisi il più possibile in diverse modalità, all'interno di un nodo ottico, per garantire il risparmio sul costo di costruzione del dispositivo. Le tre principali modalità di condivisione, mostrate anche in [4] sono:

- Shared-Per-Node (SPN)
- Shared-Per-Link (SPL)
- Shared-Per-Wavelength (SPW)

2.3.1 Shared-Per-Node (SPN)

Questa soluzione prevede che un insieme di WC sia completamente condiviso tra tutti i canali in ingresso (*input channels*). Questo schema richiede però l'uso di convertitori di tipo Tunable-input Tunable-output Wavelength Converter (TTWC) che consentano ad un pacchetto appartenenete ad una qualsiasi lunghezza d'onda di poter usare uno qualsiasi dei convertitori disponibili ed essere convertito su una qualsiasi lunghezza d'onda in uscita. Questi particolari convertitori sono molto costosi e compessi proprio perchè permettono di variare a piacimento le lunghezze d'onda di ingresso e di uscita[4].

2.3.2 Shared-Per-Link (SPL)

In questo schema i convertitori sono condivisi per fibra di uscita (*output fiber*). Anche questo schema richiede però l'uso di convertitori di tipo Tunable-input Tunable-output Wavelength Converter (TTWC) costosi e molto complessi[4].

2.3.3 Shared-Per-Wavelength (SPW)

Recentemente è stato proposto un nuovo schema di utilizzo dei convertitori, qui sono condivisi per lunghezza d'onda (da cui il nome). In questa configurazione possiamo usare dei convertitori di tipo Fixed-input Tunable-output Wavelength Converter (FTWC) meno complessi e più economici dei TTWC in quanto impostati su una determinata lunghezza d'onda di ingresso, pur mantenendo la capacità di conversione verso qualsiasi altra lunghezza d'onda di

uscita. Questa soluzione permette con un limitato numero di WC può garantire le stesse performance del caso in cui abbia a disposizione un numero di convertitori tale da coprire ogni evenienza, gsrantendo così un risparmio sul costo del dispositivo[4].

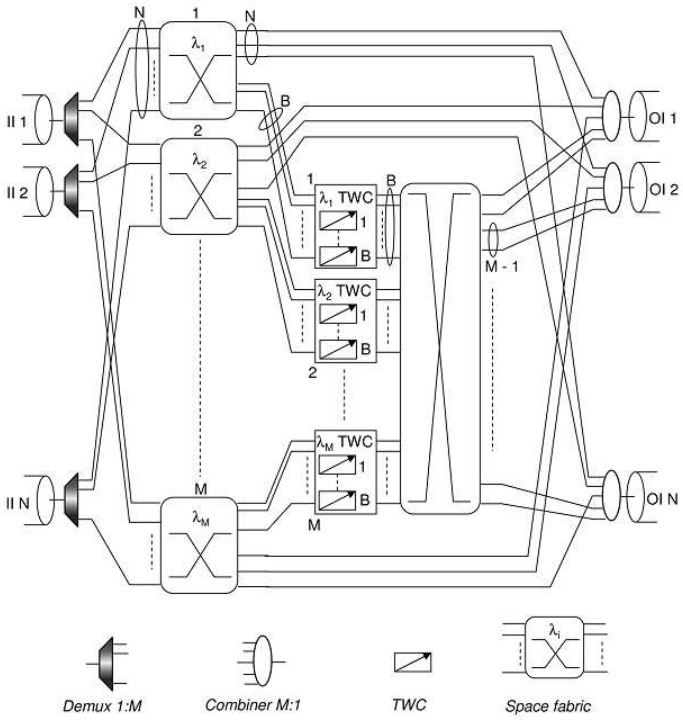

Figura 2.11: Architettura di riferimento SPW

Come si evince in figura 2.11 l'architettura base è composta da:

- N Input fiber Interfaces (II) ognuna contenente un segnale multiplexato mediante tecnica WDM (su singola fibra) con M lunghezze d'onda
- N Output fiber Interface (OI) ognuna contenente un segnale multiplexato mediante tecnica WDM (su singola fibra) con M lunghezze d'onda
- N Demodulatori 1:M che scindono il segnale WDM nelle sue M componenti (lambda1, ..., lambdam)
- N Space Fabric
- M Insiemi differenti ognugno formato da B ($<$ N) Wavelength Converter (WC) per un totale di M*B WC. Ogni insieme degli M presenti ha in ingresso segnali con la stessa lunghezza d'onda quindi può usare dei FTWC

Come accennato in precedenza questo schema garantisce una serie di vantaggi, a cominciare dall'uso di convertitori del tipo FTWC, oltre alla possibilità di organizzare la componente di switch in maniera meno onerosa in quanto è possibile usare M piccoli switch fabric per ogni lunghezza d'onda al posto di un singolo grande switching fabric necessario nell'approccio SPN. La perdita di pacchetti può avvenire per *output blocking* (se non abbiamo abbastanza lunghezze d'onda nella OI di destinazione per accomodare tutti i pacchetti ad essa indirizzati) o per *inability to convert* o inabilità a convertire il pacchetto per carenza di converitori[4].

2.4 L'architettura Broadcast-and-Select

Seppur interessanti e efficenti dal punto di vista del risparmio sui costi queste architetture non consentono una adeguata gestione di traffici differenziati per tipologia e basati su livelli di servizio. In contrapposizione alle soluzioni presentate vi è una architettura che rientra nella definizione Broadcast-and-Select (BaS). In questa architettura le informazioni vengono propagate a tutte le interfacce di uscita del nodo verso cui il percorso è ostacolato solo da alcuni dispositivi in grado di comportarsi come interruttori ottici, permettendo al segnale di passare o meno in base al loro stato [5].

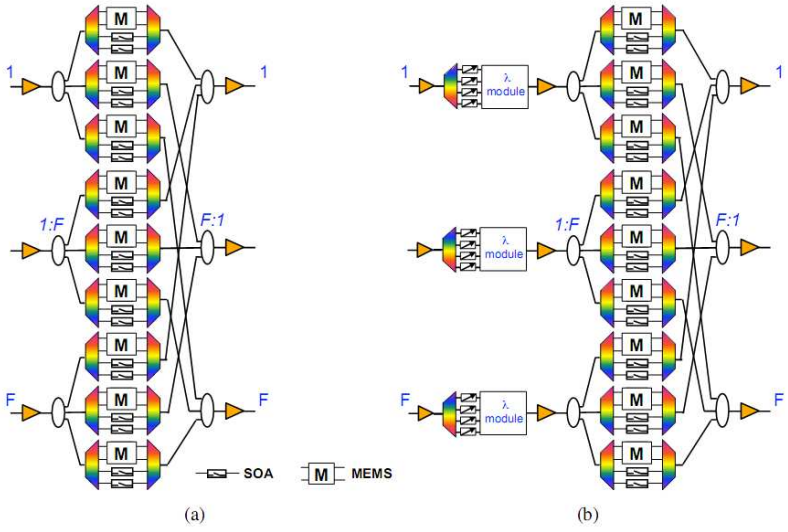

Figura 2.12: Tipologie di architettura Broadcast-and-Select.

Configurando opportunamente questi dispositivi, ovvero la matrice di commutazione ottica, secondo le direttive fornite dal piano di controllo secondo l'algoritmo di scheduling utilizzato è possibile attivare o disattivare gli inter-

ruttori ottici in modo da instrdare i pacchetti nei percorsi voluti rispettando l'ordine e le contese.

In figura 2.12 è possibile osservare la struttura di una rete di Broadcast-and-Select (BaS) tipica di un nodo centrale di una rete ottica (possibilmente anche multi-granulare). L'architettura dispone di N fibre di input/output che trasportano un segnale WDM a M lunghezze d'onda. Questo tipo di architettura è stato studiato molto in passato in una configurazione con una sola tipologia di interruttori ottici mentre qui viene proposta una architettura con dispositivi veloci (*fast*) e dispositivi lenti (*slow*) come SOA e MEMS, fornendo così buone prestazioni e garantendo la scalabilità del nodo, pur mantenendo il costo della scalabilità più basso possibile. In questo esempio F lughezze d'onda sono connesse ai dispositivi veloci, mentre M - F a dispositivi lenti. L'architettura BaS fornisce alcuni vantaggi rispetto ad altre soluzioni. Innanzitutto supporta facilmente le comunicazioni di tipo multicast e broadcast inoltre la switching fabric è implementata sfruttando i dispositivi ottici come semplici interruttori ON/OFF. Così facendo non si ha più bisogno di *switching elements* di grandi dimensioni, costosi e poco scalabili. L'architettura così proprosta è bloccante sulle lunghezze d'onda ovvero se due segnali arrivano allo switch da input differenti sulla stessa lunghezza d'onda non possono essere inoltrati alla stessa fibra di uscita a causa di collisione nello stesso canale. Questa scelta limita molto le performance del nodo, per far fronte a questo si è pensato di introdurre dei componenti in grado di effettuare la conversione di lunghezza d'onda sui segnali in arrivo in modo da risolvere quando possibile le contese sui canali. Nella figura 2.12 (b) è visibile un ulteriore pre-stadio con M Fixed-input Tunable-output Wavelength Converter (FTWC)[5]. Non limitando l'uso di dispositivi questa soluzione può risultare più dispendiosa, ma garantisce un alto supporto a quei meccanismi che consentono la trasmissione efficente di traffico altamente basato su livelli di Quality of Service (QoS).

Capitolo 3

Un modello software di router ottico

Raramente si migliora se non si ha altro modello da imitare che se stessi.

Oliver Goldsmith, scrittore e drammaturgo irlandese.
(1730 - 1774)

L'utilizzo delle reti in fibra ottica sta raggiungendo una diffusione sempre più capillare nel mondo delle telecomunicazioni. Purtroppo il punto a svantaggio di tale soluzione è l'elevato costo di un singolo nodo di commutazione che raggiunge le diverse decine di migliaia di dollari rendendone giustificato l'acquisto solo per reti metropolitane o strutture molto ampie. Da questa premessa emerge la necessità, per avere una piattaforma di studio della tecnologia, di poter in qualche modo emulare un tale dispositivo mediante una modellazione il più reale possibile, e possibilmente conforme alle Request For Comments (RFC) relative, in modo da poterlo implementare mediante degli strumenti software che, opportunamente installati su hardware dedicati, costituiscano un banco di prova con prestazioni vicine all'originale ma a costi molto più contenuti.

3.1 Architetture multi-granulari

Una risposta alla richiesta di flessibilità della rete avanzata dalle necessità delle moderne applicazioni in termini di banda, QoS e *troughput* può trovarsi nelle così dette *multi-granular optical networks*. Queste particolari reti ottiche permettono di sfruttare diverse granularità nel trasporto del traffico e sono una

soluzione chiave per servizi di alta prestazioni dinamiche di rete, nello scenario futuro di Internet. In quest'ottica l'idea di sviluppare un *framework* di emulazione per sostenere delle prove di prototipazione rapida permettendo di prendere in considerazione l'interazione tra le diverse entità del nodo, diventa lo scopo principale di questo lavoro[5]. L'architettura introdotta inoltre deve supportare il concetto di nodo ottico programmabile, ovvero definire una piattaforma software e hardware che semplifichi il controllo della rete, la sua riprogrammazione (a livello logico, come ad esempio la sua topologia) e garantire la trasparenza del servizio traducendo le tecnologie informatiche specifiche per una determinata tecnologia, rendendola indipendente sul piano astratto e logico. Questo consentirà un uso flessibile ed efficente delle risorse di rete, insieme al soddisfacimento dei bisogni delle applicazioni. La progettazione di un tale nodo programmabile permetterà la possibilità di offrire una gestione multi-servizio e multi-provider dello switch, e della relativa interfaccia con un piano di controllo migliorato per supportare servizi multi-granulari[5].

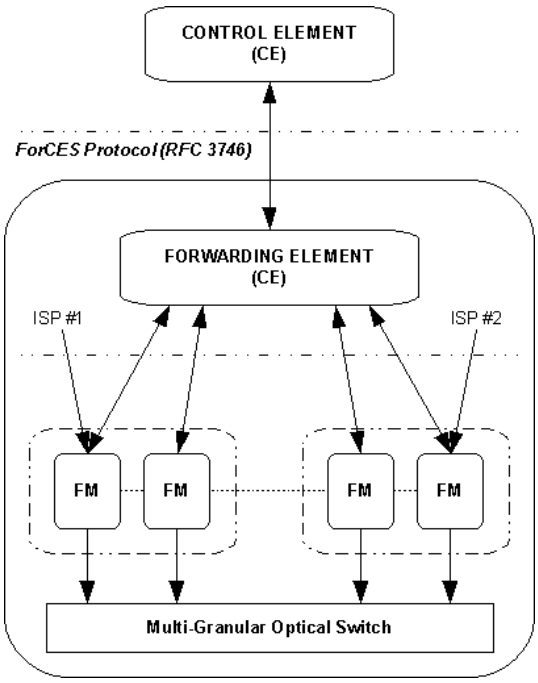

Figura 3.1: La *programmable node architecture* basata sulla raccomandazione ForCES come mostrata in [5]

Il percorso tracciato in Forwarding and Control Element Separation (ForCES) definita nella RFC 3746 rappresenta una serie di raccomandazioni in grado di definire un *framework* per la separazione nel piano di controllo di un router ottico degli elementi di controllo Control Element (CE) e di inoltro Forwarding Element (FE) definendo inoltre un protocollo standard per lo scambio delle

informazioni tra CE e FE. Questo concetto permette di rimpiazzare la comunicazione tra i moduli basata ora su standard proprietari trasformando le *network boxes* in sistemi *multi-vendor* con i sottosistemi di controllo ed inoltro sviluppabili ed evolbibili separatamente[5]. Il lavoro svolto dal gruppo di ricerca del relatore e dei correlatori di questa tesi mira ad introdurre un ulteriore modularizzazione del sistema definito nella raccomandazione ForCES nel cosidetto Forwarding Module (FM). Questo componente tende ad essere totalmente configurabile da ogni Internet Service Provider (ISP) per le proprie specifiche necessità di traffico tramite il CE e il FE. Il FE può inoltre interagire con moduli software interni al router come *meters*, *shaper* e *classifier* in modo da permettere ai dati di essere instradati attraverso il corretto FM in relazione alla classe di servizio cui appartiene il traffico.

3.2 L'architettura proposta

Presentiamo ora l'architettura base del router ottico implementato nel corso di questo lavoro. Il nodo deve essere interfacciato a un piano generico di controllo della rete (ad esempio GMPLS) tramite un'interfaccia standard che permetta al nodo di essere controllato da qualsiasi piano di controllo, in modo da garantire la distribuzione e l'interoperabilità di rete in contesti diversi. L'interazione tra piano di controllo e di gestione interna del nodo è stata affidata all'entità che si trova al livello più alto, il CE. Questo elemento elabora le richieste di servizio provenienti sui canali di segnalazione e, secondo l'applicazione specifiche esigenze di ogni richiesta in arrivo (ad esempio *long-term waveband/wavelength path establishment* o *fast sub-wavelength switching service*), negozia le risorse necessarie per la trasmissione con l'elemento dello strato intermedio, FE. Il FE è incaricato di gestire ed eseguire le operazioni logiche di inoltro internamente, verificando la disponibilità di lunghezze d'onda di conversione, risoluzione dei conflitti, di scheduling di pacchetti o flussi *burst*. L'FE è sempre a conoscenza dello stato attuale delle risorse di trasmissione, e risponde alle interrogazioni del CE secondo la capacità del nodo di soddisfare una richiesta in arrivo [5].

Tuttavia, al fine di essere il più indipendente possibile dalla tecnologia realizzativa delle risorse di inoltro, il FE deve operare su una astrazione dell'hadware di attuazione, la cui configurazione fisica è quindi delegata ai moduli di livello inferiore, o Forwarding Module (FM). Ogni FM è incaricato di guidare l'attuazione dei dispositivi hardware di uno specifico paradigma di commutazione in base alle decisioni del FE. Ad esempio, un dato FM sarà dedicato ai dispositivi di commutazione ottica lenta (ad esempio, MEMS), mentre un altro FM sarà incaricato di fissare i dispositivi di commutazione veloce (ad

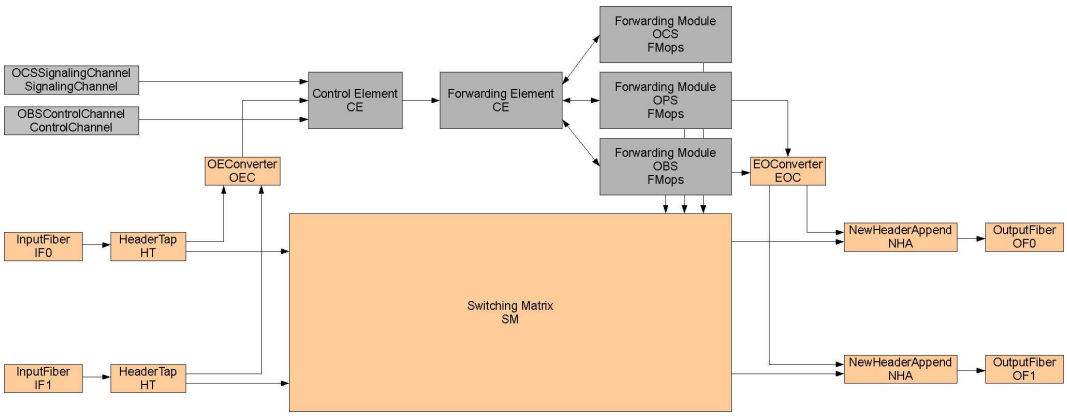

Figura 3.2: Schema generale dell'architettura proposta

esempio per la commutazione di pacchetti o flussi *burst*). Questo approccio consente una visione virtualizzata delle risorse hardware di commutazione, che può essere utilizzato dal FE quando si eseguono le funzioni di inoltro. Così i fornitori di servizi sono quindi in grado di costruire e installare i loro FM e controllarli attraverso il CE e gestire così l'interazione mediante il FE. Questo sarà incaricato di creare le tabelle di inoltro, gestire gli algoritmi di spedizione, e così via per le diverse applicazioni volute a seconda della QoS o di altre necessità. In questo modo è possibile ottenere la compartimentazione degli switch tra diversi enti, riducendo così l'ammontare dei costi attraverso la condivisione delle risorse. Gli FM interagiscono con il sistema multi-granulare per gestire i dati di spedizione secondo la granularità di trasporto richiesto gestendo lo sfruttamento dell'hardware disponibile. Finora, la progettazione dell'architettura dei nodi ottici non ha tenuto conto della interazione con il piano di controllo e, ancora più importante, della possibilità di rendere il nodo aperto e programmabile in modo flessibile. Per fare questo, è necessaria una attenta analisi di come mappare le applicazioni disponibili per le capacità di commutazione. La soluzione di nodo programmabile proposta in questo lavoro ne permetterà la configurazione dinamica e lo sfruttamento delle risorse di inoltro al fine di renderle controllabili dalle applicazioni. Di conseguenza, l'infrastruttura di rete sarà suddivisa in reti virtuali, attraverso l'astrazione delle risorse di rete, permettendo così ai diversi fornitori di condividere queste risorse e quindi trasformandola in una rete secondo il concetto multi-servizi, multi-fornitore. Gli *edge* e *core nodes* in queste reti aggregheranno il traffico in modo efficiente e gestiranno le proprie risorse in modo opportuno al fine di eseguire le operazioni di inoltro. La Switching Matrix (SM) inoltre dovrà essere gestita da algoritmini di controllo adeguati per configurare dinamicamente le risorse e fornire supproto ai servizi di rete [5].

3.2.1 L'estensione per i paradigmi OCS e OBS

L'architettura proposta, ed esplicata nel dettaglio per quanto concerne la modalità OPS è inoltre estendibile al caso in cui debba gestire, come espresso dalla sua natura, granularità di traffico diverse a livello di commutazione di circuito o flussi di traffico di tipo *burst*. Come accennato, i canali di controllo e segnalazione presenti nella figura generale della architettura[5].

3.3 Tecniche di valutazione di una architettura

Una volta definita l'architettura di commutazione in tutti i suoi aspetti, è il momento di valutare le sue prestazioni da un punto di vista logico e fisico. Per ottenere ciò è possibile seguire diversi approcci metodologici come l'analisi, la simulazione, l'emulazione o i test fisici.

L'approccio analitico è spesso difficile e oneroso da realizzare sopratutto su architetture complesse e sofisticate come quelle in esame. Testare le performance sul piano fisico significa realizzare un banco di prova o (*test-bed*) della architettura proposta, tuttavia è stato dimostrato in recenti lavori, come in [22] e [23], che con un numero elevato di porte e/o canali questi approcci sono molto costosi e spesso impossibili da testare a fondo a causa della immaturità della tecnologia ottica integrata. Inoltre con queste limitazioni, non è possibile mediante l'uso dei *test-bed* verificare le prestazioni logiche (ad esempio la perdita di pacchetti) dell'architettura.

Quindi in questi casi occorre testare l'architettura mediante un approccio diverso, come la simulazione o l'emulazione software.

Per simulazione si intende la creazione di un modello della realtà che si desidera studiare che consenta di valutare e prevedere lo svolgersi dinamico di una serie di eventi susseguenti all'imposizione di certe condizioni da parte dell'analista o dell'utente. L'emulazione invece nel senso più generale possibile accepisce alla caratteristica di duplicare le funzioni di un determinato sistema su un secondo sistema differente dal primo ovvero permette l'esecuzione di una serie di comportamenti definiti un ambiente (traffico ottico) diverso da quello sul quale l'emulatore viene eseguito (ambiente eletronico). Analizziamo ora in dettaglio le caratteristiche di queste due soluzioni mettendo in risalto le potenzialità ed i limiti di ognuna in modo da motivare la scelta effettuata.

3.3.1 La simulazione

La simulazione è il modo più semplice e diretto per implementare funzionalità di una data architettura permettendo di ottenere risultati realistici, quando

le funzionalità degli switch ottici siano adeguatamente definite e considerate nello strumento di simulazione. Per questo motivo, numerose attività di ricerca in questo campo vengono effettuate tramite l'uso di simulatori ad-hoc o facenti parte di un *framework* di simulazione (come OPNET, NS2, ...). D'altra parte il suo utilizzo comporta un dispendio notevole in termini di tempo, specialmente in presenza di eventi rari di campionamento. Per ottenere risultati significativi e ragionevoli, infatti, i parametri di simulazione devono essere configurati correttamente, il che spesso non è un compito facile da eseguire. Inoltre, le interazioni tra il controllo (vale a dire le decisioni di *forwarding*) e il *set-up* dell'hardware di commutazione non è effettivamente messo in evidenza attraverso la simulazione, a causa della mancanza di conoscenza delle caratteristiche dell'hardware reale. Infine, se il traffico in ingresso simulato non è realistico e si discosta molto dagli scenari applicativi reali, questo potrebbe generare risultati fuorvianti. Pertanto, la simulazione in sé non è sufficiente per ottenere una buona comprensione delle funzionalità di un nodo ottico.

3.3.2 L'emulazione

Per questi motivi si è pensato che l'emulazione software di dispositivi di commutazione ottici e dei relativi moduli di controllo collegati potrebbe rappresentare un buon modo per fornire una valutazione delle prestazioni sia fisiche e logiche. I software router consentono infatti la modularizzazione della architettura e l'emulazione dei dispositivi ottici utilizzati per costruire il l'hardware commutazione (sia dal punto di vista logico che fisico). I moduli sono poi collegati gli uni agli altri per emulare l'architettura del nodo. Questo approccio permette di valutare, osservare e testare le interazioni tra i diversi moduli, migliorando la flessibilità rispetto all'approccio della simulazione. Inoltre l'architettura può essere testato su un *framework* di emulazione piuttosto che dover implementare *test-bed* complessi, costosi e difficili da gestire. I principali vantaggi della emulazione per quanto riguarda la simulazione sono:

- **Modularità del documento**: una volta definiti, i moduli possono essere aggiunti, rimossi, collegati e scambiati per ottenere diverse architetture;
- **Chiarezza punto di interazione**: le interazioni tra i diversi moduli sono esplicitate dalle connessioni tra gli stessi che possono essere prese in considerazione ed analizzate;
- **Testabilità dei moduli**: i moduli software dedicati al controllo dello switching fabric possono essere testati, anche se il costoso hardware che li costituisce nella realtà non è disponibile;

- **Facile modifica del traffico in ingresso**: il traffico in ingresso può essere facilmente modificato cambiando la sorgente.

Per raggiungere questi obiettivi ed essere in grado di fornire un adeguato livello di flessibilità e capacità di integrazione con la finalità, per questi sistemi di poter essere utilizzati in contesti reali l'autore ritiene che l'emulazione si adatta bene ai requisiti di prova di una intera architettura nodo.

3.4 Il software di programmazione *Click!*

Ora introdurremo brevemente e negli aspetti in uso al nostro lavoro la piattaforma software su cui è basata l'implementazione della architettura voluta, ovvero il *Click!* Modular Router. Cenni ulteriori e un breve approfondimento del *framework* è presente in appendice A.4.1 a questo testo.

3.4.1 Introduzione al *Click!*

Il *Click!* è una architettura software open-source modulare, orientata alla realizzazione di una vasta gamma di dispositivi come *router*, processori di pacchetti, sorgenti di traffico, *Ethernet switch*, *firewall* ... basata su piattaforma GNU/Linux e sviluppata dal Massachusetts Institute of Technology (MIT).

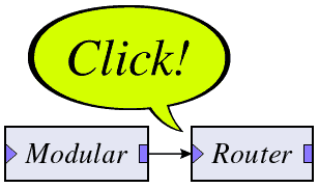

Figura 3.3: Logo del software Click Modular Router

Un qualsiasi dispositivo *Click!* è modellato esclusivamente attraverso l'aggregazione di moduli di elaborazione dei pacchetti chiamati *elementi* e non esiste ulteriore astrazione per un componente (del dispositivo) oltre a questa. Gli elementi inoltre sono collegati tra loro mediante delle linee orientate che rappresentano il flusso dei pacchetti che li attraversa dette *connessioni*. Ogni elemento implementa semplici funzioni come ad esempio la classificazione del traffico o l'accodamento piuttosto che lo scheduling o l'interfacciamento con i dispositivi di rete. Un insieme di elementi connessi con più linee orientate rappresenta una *configurazione*, ovvero il modello del dispositivo che vogliamo simulare[24][17].

3.4.2 Elementi, connessioni e pacchetti

Diamo ora una breve descrizione dei costrutti base del *Click!* e delle loro funzionalità al fine di rendere più chiara la comprensione del lavoro svolto.

Elementi

Un elemento è un componente software che rappresenta un'unità di elaborazione del traffico (inteso ad esempio come pacchetti dati) ed esegue concettualmente semplici calcoli, come ad esempio decrementare il campo Time-to-live (TTL) di un pacchetto, piuttosto che calcoli complessi, come il routing IP. In generale essi esaminano o modificano i pacchetti in un certo modo. Ogni elemento è un oggetto C++ che può mantenere una sua autonomia. Dispositivi di processamento, tabelle di instradamento, gestione delle code, conteggi e quant'altro, sono tutte funzioni implementate dagli elementi. Gli elementi sono dunque definiti come istanze di classi C++ in cui è specifica la struttura dei dati che possono essere trattati dall'elemento stesso e il suo comportamento (quante porte avrà, quali handlers supporterà e come elaborerà i pacchetti). In C++ ogni elemento corrisponde ad una sottoclasse della struttura *Element*. Inoltre ogni elemento può avere un numero arbitrario di porte di ingresso e uscita collegate ad altri elementi mediante dell connessioni. Ogni connessione collega una porta di uscita di un elemento con una porta di entrata di un altro. Il numero di porte di un elemento può essere fisso, oppure può dipendere da una stringa di configurazione, oppure da quante porte sono usate nella particolare configurazione. Particolare caratteristica degli elementi è la definizione degli *Handler*. Questi sono modalità di interfaccia esportate a livello di utente piuttosto che agli altri elementi della configurazione router. Ad esempio l'elemento *Queue* ha un *handler* che riporta la sua lunghezza corrente come una stringa ASCII decimale, mentre l'elemento *Counter* mette a disposizione un *handler* che permette all'utente di conoscere il valore corrente del suo contatore[24][17].

Pacchetti *Click!*

I pacchetti *Click!* rappresentano le particelle elementari della comunicazione, trasportando le informazioni che vengono eleborate dagli elementi. Durante il funzionamento del dispositivo mappato nella configurazione, i pacchetti passano da un elemento all'altro attraverso collegamenti chiamati connessioni. Un pacchetto *Click!* consiste in una serie di annotazioni e in un puntatore al reale campo dati del pacchetto IP.

L'intestazione del pacchetto punta ai dati (rappresentati in figura 3.4 dal pacchetto ottico), e grazie a questo meccanismo molte intestazioni possono

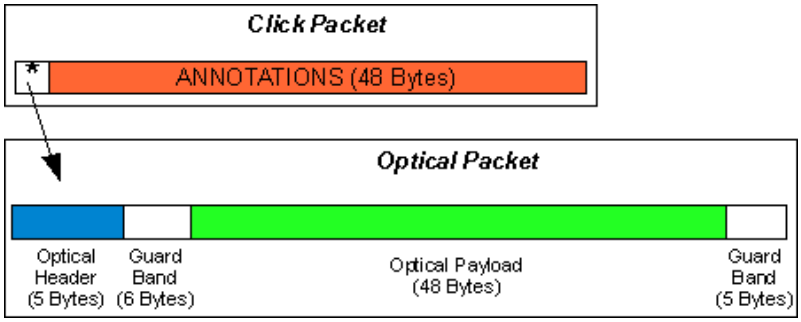

Figura 3.4: Schema rappresentativo di un pacchetto *Click!*

condividere gli stessi dati. Quando si copia un pacchetto, ad esempio tramite l'elemento *Tee*, il *Click!* produce una nuova intestazione che condivide lo stesso pacchetto dati. I pacchetti di dati condivisi sono detti perciò *copy-on-write* mentre le intestazioni invece non lo sono mai, così la loro modifica non provoca mai una copia.

Oltre al puntatore ai dati l'intestazione contiene un certo numero di annotazioni, le quali possono essere condivise con il sistema operativo Linux, oppure mediante specifiche del linguaggio. Alcune annotazioni contengono informazioni indipendenti dai dati (per esempio il tempo in cui il pacchetto è arrivato), mentre altre memorizzano informazioni riguardanti i dati. Le annotazioni sono memorizzate nell'intestazione del pacchetto in un ordine fisso e statico mentre i dati del pacchetto sono memorizzati in un singolo *buffer* di memoria[24][17].

Connessioni

Ogni connessione rappresenta un percorso possibile per il trasferimento dei pacchetti e collega una porta di uscita di un elemento con una porta di ingresso di un altro, ed ognuna rappresenta un possibile percorso per il trasferimento dei pacchetti tra gli elementi. In una configurazione *Click!* in esecuzione le connessioni sono rappresentate come puntatori agli oggetti elemento e il passaggio dei pacchetti lungo una connessione è implementato da una singola chiamata di una funzione virtuale. Graficamente le connessioni vengono rappresentate come frecce che collegano una porta sorgente ad una porta destinazione, indicando la direzione del flusso di pacchetti[24][17].

3.4.3 Il linguaggio e le configurazioni

Gli ultimi due tasselli del *framework Click!* , utili ad una maggiore comprensione del lavoro di implementazione svolto, sono mostrati di seguito.

Il linguaggio

Alla base della programmazione *Click!* vi è un linguaggio dichiarativo, ovvero un linguaggio che descrive semplicemente il grafo relativo ad una configurazione (a differenza dei linguaggi di *script*, come per esempio i file .tcl di Network Simulator 2 (NS2)). I linguaggi dichiarativi hanno il vantaggio della leggibilità, ma soprattutto possono essere analizzati e modificati in maniera molto più semplice di quanto invece non consentano i linguaggi imperativi.

Il linguaggio *Click!* è semplice, dato che comprende un numero ridotto di costrutti utilizzabili, preferendo implementare delle estensioni del linguaggio attraverso elementi che avessero degli scopi specifici. Questa scelta limita i meccanismi ed i costrutti che possono essere utilizzati in fase di programmazione, ma consente di ottenere una grande flessibilità. I programmi scritti in linguaggio *Click!* e le configurazioni grafiche sono due strutture equivalenti.

Come abbiamo visto ciascun elemento appartiene ad una classe di elementi, la quale è specificata dal nome ed opzionalmente da una stringa di configurazione. Gli elementi sono connessi attraverso le loro porte d'ingresso e d'uscita. Nel linguaggio specifico del *Click!* , le porte sono distinte attraverso l'uso di numeri, mentre gli elementi sono distinti utilizzando un nome. Ciascun elemento in una configurazione possiede un unico nome, che un utente può specificare in maniera opzionale. Tali elementi individuano e differenziano i vari elementi durante il processo di analisi (anche sintattica eseguita in fase di compilazione), ed permettono inoltre al singolo utente, o ad altri programmi, di accedere ad un particolare elemento, anche in fase di esecuzione (lancio della configurazione).

Definizione di una configurazione

Una configurazione è descritta mediante un semplice file di testo che rispetti la sintassi del linguaggio *Click!* . Essa può essere pensata come un grafo orientato in cui gli elementi ne rappresentano i vertici. Una configurazione è solitamente divisa in due parti:

- **Dichiarazioni**: sezione in cui si definiscono gli elementi e se ne specifico i parametri.
- **Connessioni**: sezione in cui viene specificato come connettere gli elmenti sopra definiti

Una qualsiasi configurazione *Click!* è costituita quindi una serie di componenti elementi, scelti dall'utente in base alle loro caratteristiche e peculiarità

e in base alle finalità del modello, uniti mediante delle connessioni. Per implementare delle estensioni o dei comportamenti specifici per gli elementi si possono scriverne di nuovi, oppure comporre più elementi esistenti nel modo opportuno[24][17].

3.5 Implementazione software della architettura

Mostreremo ora in dettaglio una prima implementazione dell'architettura di tipo Broadcast-and-Select (BaS) con estensione multi-granulare esposta precedentemente, dettagliando gli aspetti implementativi d questa prima versione. Dato il lavoro di ricerca e documentazione prima, di sviluppo e testing poi si è deciso di portare a termine una prima versione della architettura che se pur incompleta in alcuni aspetti, fosse funzionale e in grado di restituire dei risultati da poter analizzare e confrontare. Inoltre l'implementazione proposta sarà facilmente estendibile nelle componenti momentanemante trascurate, non pregiudicando quindi il lavoro fin qui svolto. In particolare ci si è concentrati sulla realizzazione di una architettura di tipo Broadcast-and-Select (BaS) multi-granulare di cui si è implementato il solo paradigma di commutazione OPS tralasciando per ora la commutazione ottica di circuito e a *burst*. Per compensare a questa carenza si è puntato ad una relizzazione di una rete OPS di tipo *slotted* sincrona, in quanto il caso asincrono, più generale, può essere da questa facilmente derivato.

3.5.1 Emulazione della architettura

Una interessante possibilità per costruire un banco di prova programmabile per *router* ottici, in grado di emularne il funzionamento, è rappresentato come abbiamo visto dal *framework Click! Modular Router*[17]. Questo approccio è stato recentemente dimostrato efficace per la misurazione e il controllo programmabile dei router nei lavori [25][26][27].

La nuova sfida è rappresentata dalla progettazione di una estensione del *framework* al supporto di complesse funzioni di routing, organizzate in moduli, che interagiscono tra di loro per mezzo di un protocollo adatto come specificato per esempio nella direttiva Forwarding and Control Element Separation (ForCES).

3.5.2 Analisi ed emulazione dei livelli di potenza

Degna di nota è l'estensione del modello per l'emulazione dei livelli di potenza dei segnali e dei relativi andamenti all'interno della architettura. L'emulazione

infatti, in seguito alla generazione casuale di un livello di potenza discretizzato per ogni segnale in ingresso è in grado poi di attribuire una attenuazione della potenza stessa per ogni componente passivo attraversato dal segnale, su una determinata lunghezza d'onda, calcolata mediante l'elaborazione di formule ideali per l'attenuazione delle varie tipologie di dispositivi. Specularmente il modello è in grado di emulare guadagni e equalizzazioni del livello di potenza del segnale nel caso questi attraversi componenti attivo come, ad esempio, i SOA. Seppur implementata ad un livello iniziale questa estensione sarà in futuro in grado di presentare un resoconto dettagliato sull'andamento del consumo di potenza e sull'entità dei valori in gioco, indice di notevole interesse per le archiettture in esame.

3.5.3 Implementazione basata sul software *Click!*

Una possibile configurazione che implementi l'architettura programmabile mostrata nei paragrafi precedenti, con riferimento ad un nodo ottico che implementi i paradigmi di commutazione ottica a circuito, *burst* e pacchetto è mostrato in figura 3.5.

Il software *Click!* ci permette di definire dei file di configurazione altamente modulari in grado di realizzare i passi elementari della architettura. Ad esempio la molteplicità di lunghezze d'onda è simulata usando la caratteristica Paint Annotation la quale ci permette di marcare i dati e distinguere le varie lunghezze d'onda. Lo schema considerato, può essere suddiviso concettualmente su due piani diffrenti:

- **Control Plane**: costituito dagli elementi più scuri, identifica il piano in cui si prendono le decisioni in termini di controllo ed inoltro del traffico.
- **Data Plane**: costituito dagli elementi più chiari, identifica il piano in cui viaggia il traffico dati.

Gli elementi più scuri rappresentano la realizzazione dei moduli del piano di controllo, cioè il Control Element (CE), Forwarding Element (FE) e i tre Forwarding Module (FM) uno per ogni paradigma di commutazione. Il CE gestisce la prima parte del piano di controllo del nodo, avendo cura delle richieste provenienti dai canali di segnalazione e controllo fuori banda dei paradigmi OCS e/o OBS e di segnalzione in banda dell'OPS (cioè l'intestazione dei pacchetti). La matrice di commutazione può essere implementata utilizzando degli elementi *Click!* opportunamente personalizzati che emulano le caratteristiche logiche e fisiche, in termini di configurazione e ritardi offerti dai diversi dispositivi, attenuazione del segnale ottico, segnale-rumore e disponibilità della risorsa. Questo approccio potrebbe costituire la base per una futura integrazione

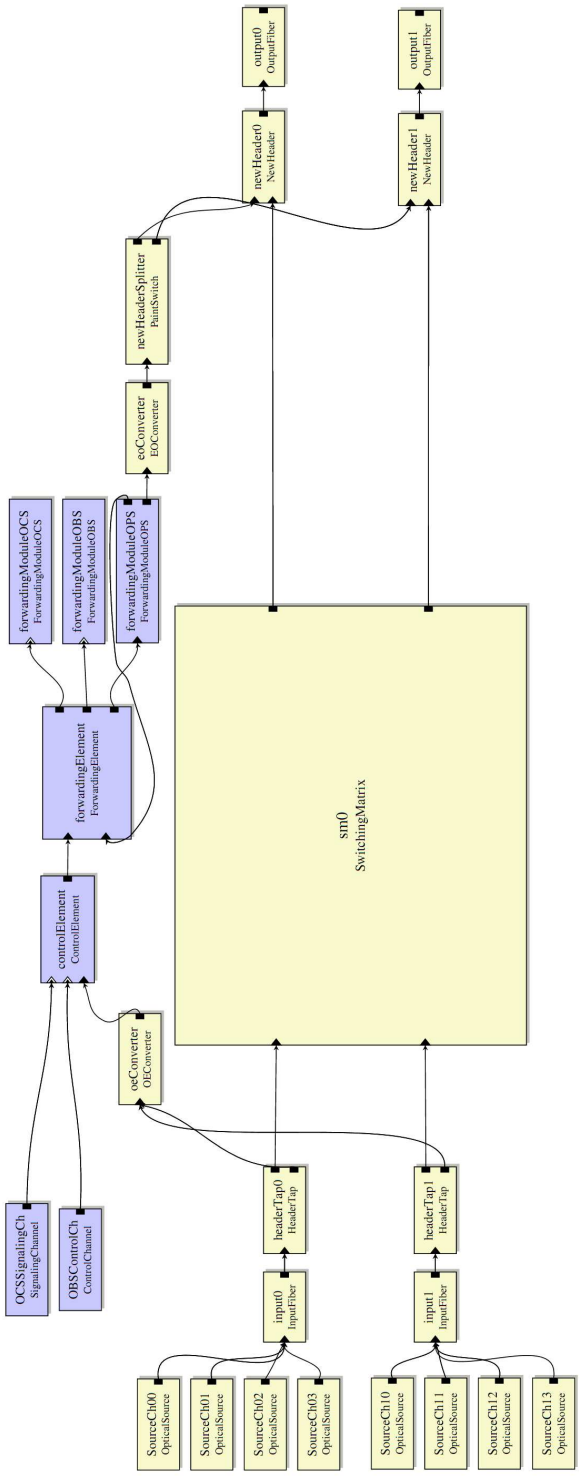

Figura 3.5: Implementazione software della architettura proposta mediante software *Click!*

di diverse funzionalità sviluppate per il controllo e la trasmissione, infatti, la modularità garantisce che i diversi moduli possano essere rapidamente sostituiti con quelli nuovi e testati molto facilmente. L'analisi comparativa del router ottico programmabile qui modellato può inoltre essere emulata fornendo un traffico reale da parte di generatori ad hoc già disponibili.

Vediamo ora in dettaglio i componenti della architettura suddivisi sui vari piani logici, introducendono le caratteristiche e le funzionalità.

3.6 Control Plane

Rappresentato dagli elementi di colore più scuro, identifica il piano in cui si prendono le decisioni in termini di controllo ed inoltro del traffico. E' quindi composto dal Control Element (CE), Forwarding Element (FE) e dai tre Forwarding Module (FM).

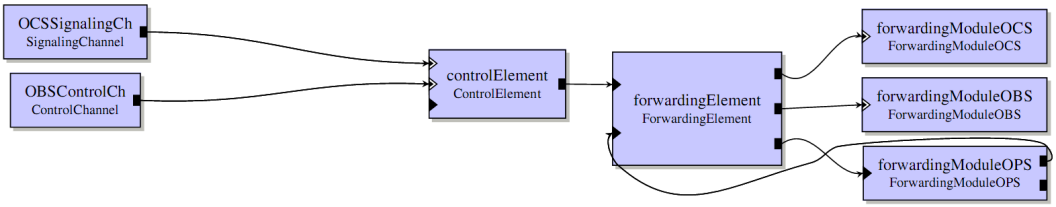

Figura 3.6: Il piano di controllo dell'architettura in ambiente *Click!*

3.6.1 OCS Signaling Channel

Questo elemento realizzerà in futuro, quando l'architettura sviluppata sarà estesa a livello multi-granulare, il canale dedicato alla segnalazione per l'instaurazioni di comunicazioni basate sul paradigma a commutazione di circuito.

3.6.2 OBS Control Channel

Anche questo elemento realizzerà in futuro, quando l'architettura sviluppata sarà estesa a livello multi-granulare, il canale di controllo per l'instaurazioni di comunicazioni basate sul paradigma di comunicazione *burst*.

3.6.3 Control Element (CE)

Il CE gestisce le richieste in arrivo dai canali di segnalazione fuori banda di tipo OCS e/o OBS e le richieste in banda di tipo OPS (contenute negli header

dei pacchetti). Il CE dopo aver identificato le necessità associate alle richieste in arrivo, invia la richiesta al FE.

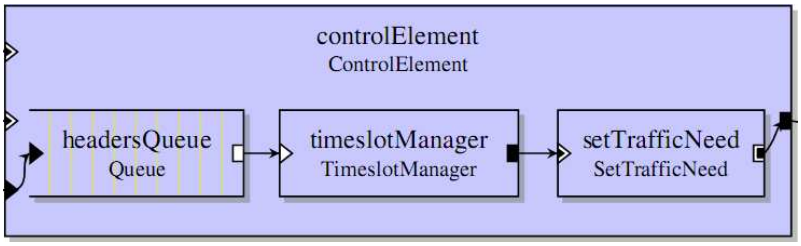

Figura 3.7: Elemento composto ControlElement

HeadersQueue

Questo elemento realizza una coda FIFO di bufferizzazione degli header convertiti nel dominio elettronico che si accodano per essere processati. L'estrazione da questa coda avviene in modalità *pull* da parte dell'elemento successivo, il TimeslotManager.

TimeslotManager

Il TimeslotManager estrae il valore del secondo byte dell'intestazione del pacchetto ottico. Questo valore esprime il timeslot cui il pacchetto appartiene e viene memorizzato così nella annotazione *Click!* relativa.

SetTrafficNeed

Questo componente realizzerà in futuro la cattura delle informazioni relative alle necessità richieste dal traffico in esame. Questo aspetto, legato alla multigranularità del traffico è momentaneamente impostato per generare delle informazioni relative alle necessità che inidirizzino il traffico sempre e solo al modulo di inoltro destinato al paradigma OPS.

3.6.4 Forwarding Element (FE)

Il FE gestisce le operazioni logiche di forwarding in accordo alle richieste che gli arrivano dal CE e dai messaggi di notifica provenienti dai FM. Il FE è concepito come indipendente dall'hardware quindi prende le sue decisioni indipendentemente dalle risorse di switching disponibili a livello hardware garantendo la separazione tra piano di controllo e piano dati. A questo punto FE invia la richiesta allo specifico FM in accordo al traffic need impostato dal CE con il messaggio di configurare opportunamente la matrice se possibile.

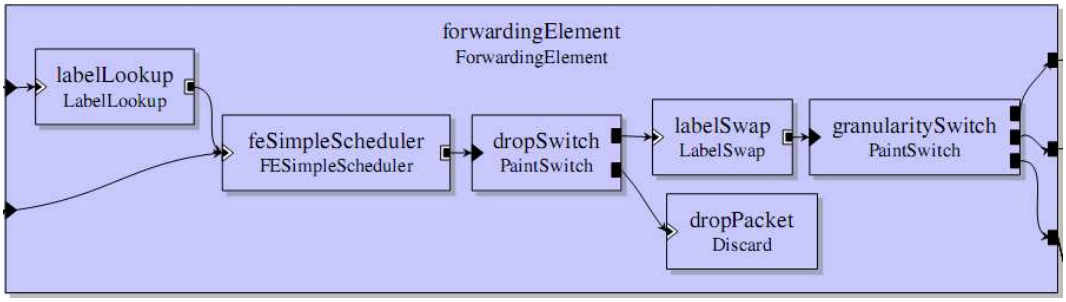

Figura 3.8: Elemento composto *Click!* ForwardingElement

Label Lookup (LL)

Questo elemento legge il primo byte dell'header ottico del pacchetto e lo interpreta come la label di destinazione del traffico. Questa viene dunque inoltrata ad una struttura interna che implementa la *Forwarding Table* e così propriamente chiamata. L'interrogazione restituisce le informazioni sulla nuova destination label che l'header dovrà assumere una volta processato e sulla interfaccia di uscita che il pacchetto dovrà seguire per raggiungere la destinazione espressa. Queste informazioni veranno scritte nei relativi campi annotazione.

FESimpleScheduler

Questo elemento realizza in dettaglio l'algoritmo di scheduling nella sua parte indipendente dall'hardware. Infatti recependo le specifiche espresse nella direttiva ForCES in questo componente non si ha conoscenza dell'hardware a disposizione e quindi i controlli sullo scheduling dei pachetti devono avvenire senza conoscenza diretta dello status della matrice di commutazione. Nella versione implementata questo componente conta semplicemente i pacchetti schedulati ed effettivamente inoltrati per ogni fibra di uscita. Se, per il corrente timeslot, per l'interfaccia di destinazione che il pacchetto in esame vuole raggiungere sono già stati inoltrati W pacchetti, ciò vuol dire che non esiste possibilità alcuna che il pacchetto, anche convertendo su un altra ulteriore lunghezza d'onda, possa raggiungere l'uscita. Viene dunque scartato, incrementando l'opportuno contatore. Altrimenti, se per il corrente timeslot, per l'interfaccia di destinazione che il pacchetto in esame vuole raggiungere sono stati inoltrati un numero minore di W pacchetti, ciò vuol dire che esiste la possibilità che il pacchetto, magari dopo un opprotuna conversione su un altra lunghezza d'onda in caso di contesa, possa raggiungere l'uscita. Questa eventualità però è demandata al ForwardingModule che è a conoscenza dell'hardware e del suo stato corrente e quindi più qualificato a determinare il buon esito della schedulazione. Le informazioni sul numero di pacchetti usciti

per ognuna delle fibre di output è fornito al FESimpleScheduler dal ForwardingModule mediante l'opportuna connessione in retroazione che realizza così una delle linee di comunicazione del piano di controllo.

Label Swap (LS)

Questo elemento riscrive il primo byte dell'header ottico del pacchetto con la nuova destination label ottenuta precedentemente dal Label Lookup (LL).

3.6.5 Forwarding Module (FM) OPS

Il FM decide quale richiesta proveniente dal FE può essere soddisfatta e quale no in accordo alla disponibilità di risorse hardware e l'occupazione delle risorse corrente per richiesta ricevuta. Se risulta possibile soddisfare la richiesta l'FM pilota fisicamente gli switch hardware in accordo alle richieste e ritorna il successo della operazione al FE. In caso non risultasse possibile soddisfare la richiesta l'FM ritorna un messaggio di insuccesso della operazione al FE che può decidere, qualora fosse possibile, di inoltrare la richiesta ad un altro FM (es. *fast-to-slow*). Il controllo dei dispositivi hardware è reso possibile agli FM in ambiente *Click!* dall'uso degli handlers, non visibili in figura che permettono ad un elemento di modificare la configurazione di un altro elemento (es. la Switching Matrix (SM)) a *run-time* in modo che sia possibile emulare il comportamento della logica di controllo di un nodo ottico.

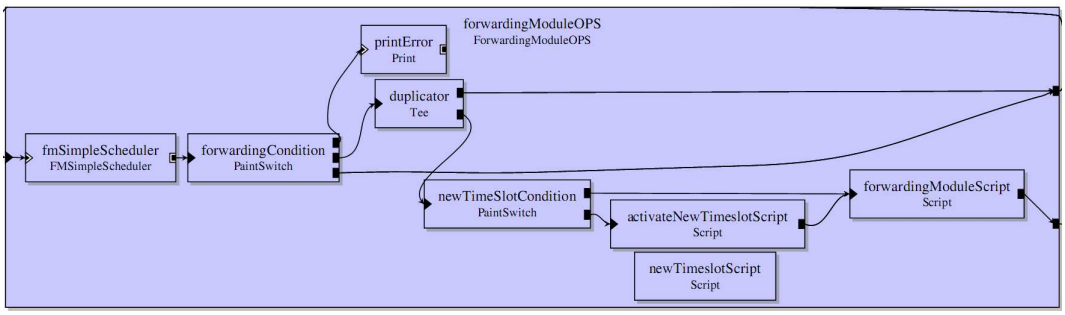

Figura 3.9: Elemento composto *Click!* ForwardingModuleOPS

FMSimpleScheduler

Questo elemento realizza in dettaglio l'algoritmo di scheduling nella sua parte dipendente dall'hardware. Recependo le specifiche della direttiva ForCES in questo componente si è finalmente a conoscenza dell'hardware a disposizione e

quindi i controlli sullo scheduling dei pachetti possono avvenire con la conoscenza diretta dello status della matrice di commutazione. Nella versione implementata questo componente controlla se per il pacchetto da schedulare esiste un canale libero per la sua lunghezza d'onda, o se è possibile convertirlo su un altra ulteriore lunghezza d'onda per raggiungere l'uscita. Se esiste un incastro possibile, tra convertitori, SOA e fibra in uscita il pacchetto è segnalato al FE come schedulato e viene posto in ingresso allo script di controllo dell'attuazione. Altrimenti il pacchetto viene scartato e viene comunque inviata un opportuna segnalazione al FE che, in un'implementazione futura, potrebbe prevedere di reinviare il pacchetto ad un differente FM.

NewTimeslotScript

Questo script di controllo, resetta la matrice ad ogni nuovo timeslot. Se il pacchetto che esce dal FM è il primo pacchetto di un nuovo timeslot questo script provvede a disabilitare i Wavelength Converter (WC) ed i SOA in modo da ripristinare la matrice per l'inoltro dei pacchetti nel corrente quanto temporale.

FMScript

L'FMScript è il cuore del meccanismo di attuazione dei dispositivi della matrice di commutazione. Leggendo gli handler dell'elemento FMSimpleScheduler è in grado di attivare il Wavelength Converter (WC) relativo al pacchetto, sulla lunghezza d'onda impostata dallo scheduler del FM, e il relativo SOA.

3.7 Data Plane

A questo livello nell'architettura vi sono tutti i componenti fisici del nodo e che sono fisicamente attraversati dal traffico dati.

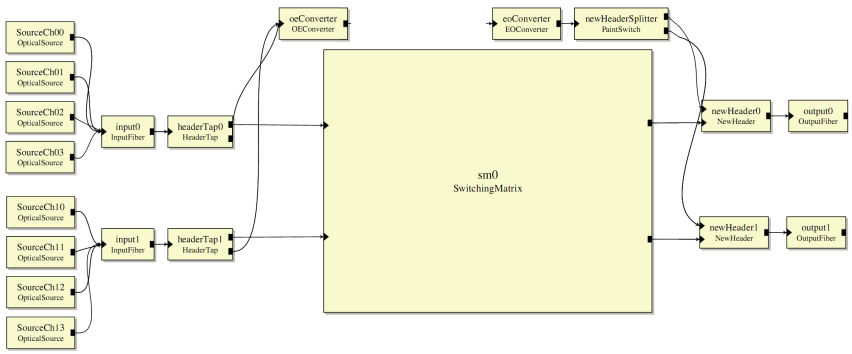

Figura 3.10: Il piano dati dell'architettura in ambiente *Click!*

3.7.1 Optical Source

Questo componente, collegato ad una socket sulla macchina di test, rappreseta la sorgente di pacchetti per una determinata lunghezza d'onda, che poi si pone in ingresso alla fibra di input.

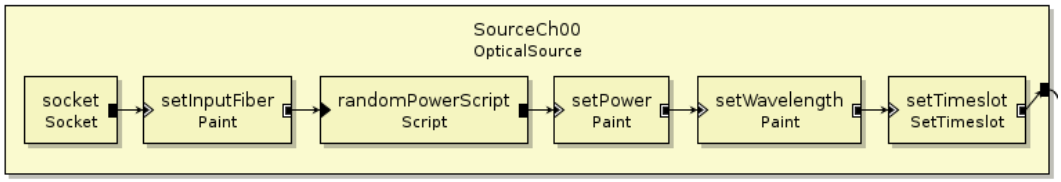

Figura 3.11: Elemento *Click!* OpticalSource

Socket

Elemento della classe Socket del *Click!* permette di ricevere i pacchetti da una socket UDP specificando un indirizzo ed una porta validi.

setInputFiber

Elemento della classe Paint del *Click!* ci permette di marcare nelle annotazioni il byte relativo alla fibra di ingresso del pacchetto.

randomPowerScript

Elemento della classe Script del *Click!* ci permette di calcolare un valore casuale di potenza per un range impostato.

setPower

Elemento della classe Paint del *Click!* ci permette di marcare nelle annotazioni il byte relativo alla potenza del segnale.

setWavelength

Elemento della classe Paint del *Click!* ci permette di marcare nelle annotazioni il byte relativo alla lunghezza d'onda di appartenenza del pacchetto.

setTimeslot

Elemento della classe Paint del *Click!* ci permette di marcare nelle annotazioni il byte relativo al numero di timeslot di appartenenza del pacchetto.

3.7.2 Input Fiber (IF)

Questo elemento modella la fibra di ingresso. Al suo interno sono presenti vari elementi OpticalSource in grado di connettere la configurazione a più sorgenti di provenienza dei pacchetti nel caso si voglia testare il modello proposto generando internamente i pacchetti, o ascoltando una *socket* UDP in modalità *userlevel* o catturando i pacchetti da una interfaccia di rete se la configurazione è installata come modulo *kernel*.

3.7.3 HeaderTap (HT)

Questo elemento, analizza i pacchetti nel dominio ottico e ne separa l'header dal payload. L'header è inviato in ingresso al CE dove rappresenta la richiesta in banda di tipo OPS, mentre il payload è posto in ingresso alla SM.

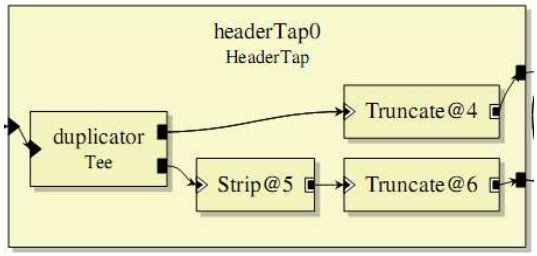

Figura 3.12: Elemento *Click!* HeaderTap

3.7.4 OEConverter

Questo elemento simula la conversione dell'header ottico del pacchetto nel dominio elettronico.

3.7.5 Switching Matrix (SM)

Per semplicità nella implementazione è stata impostata una matrice di tipo Broadcast-and-Select (BaS) che consenta di connettere due N = 2 fibre ottiche in ingresso a M = 2 fibre ottiche in uscita. Ogni fibra ottica trasporta poi un segnale WDM a W = 4 canali o lunghezze d'onda. I vantaggi proposti da questa architettura per la matrice di commutazione, rispetto ad altre soluzioni sono stati presentati nel capitolo precedente. Brevemente ricordiamo che questa configurazione ci permette di supportare comunicazioni di tipo *multicast* e *broadcast*, permette l'utilizzo di disositivi ottici (ad es. Semiconductor Optical Amplifier (SOA)) come interruttori di tipo ON/OFF permettendo così di

evitare l'uso di *switching elements* di grandi dimensioni, costosi, scarsamente prestazionali e poco scalabili.

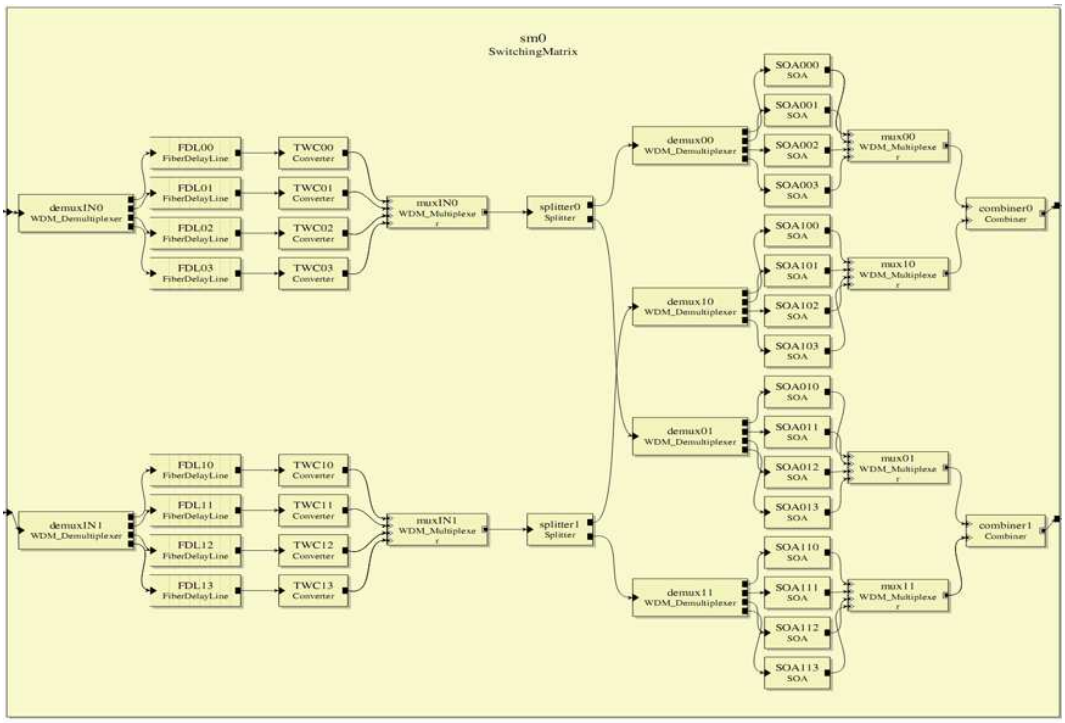

Figura 3.13: Dettaglio della Switching Matrix

WDM Demux

Questo componente *Click!* emula un dispositivo ottico passivo in grado di scindere un segnale ottico WDM nelle sue componenti per lunghezza d'onda. A livello di potenza l'elemento è impostato per attenuare il segnale secondo la formula ideale:

$$attenuazione = 1,2log_2W$$

dove W indica il numero di lunghezze d'onda componenti la fibra.

Fiber Delay Line (FDL)

Questo componente *Click!* emula un dispositivo ottico in grado di ritardare la luce in esso intrappolata per un valore determinato di tempo. L'uso di tali componenti permette di ovviare al problema della mancanza di dispositivi di memorizzazione com le RAM in ambiente elettronico.

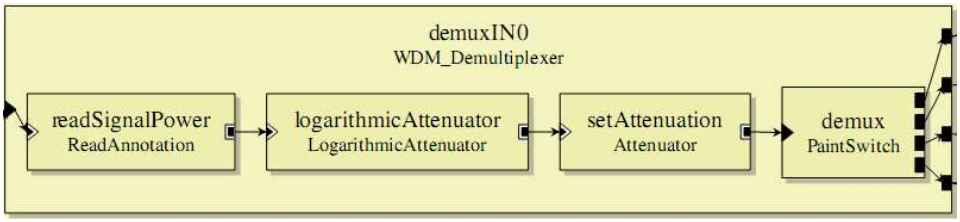

Figura 3.14: WDM Demultiplexer

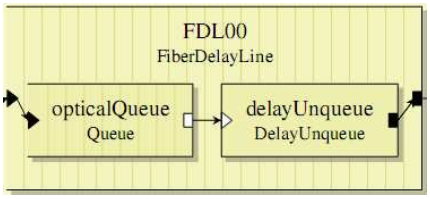

Figura 3.15: Fiber Delay Line

Converter

Questo componente rappresenta uno stadio per la conversione della lunghezza d'onda dei pacchetti, basata su convertitori di tipo Fixed-input Tunable-output Wavelength Converter (FTWC) che permette di incrementare le performance in termini di *packet loss* risolvendo le contese. Quando due o più pacchetti contendono per la stessa fibra di uscita sulla stessa lunghezza d'onda, uno è inoltrato direttamente mentre gli altri sono convertiti su un'altra lunghezza d'onda libera.

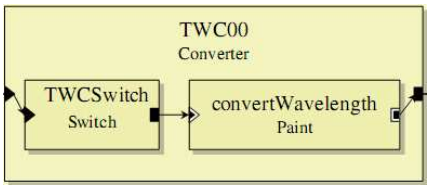

Figura 3.16: Tunable Wavelength Converter

Splitter

Questo componente emula un dispositivo ottico in grado di replicare l'ingresso su un numero variabile di uscite. Questa replicazione introduce però una attenuazione del segnale proporzionale al numero di porte in uscita di un valore logaritmico. A livello di potenza dunque l'elemento è impostato per attenuare il segnale secondo la formula ideale:

$$attenuazione = 10log_{10}N$$

dove N indica il numero di fibre su cui il segnale è replicato.

Semiconductor Optical Amplifier (SOA)

Questo componente emula un Semiconductor Optical Amplifier (SOA) ovvero un dispositivo ottico in grado di comportarsi come un interruttore, quindi permettendo o meno alla luce di attraversarlo, e in caso positivo è anche in grado di amplificare, o meglio equalizzare, il segnale nel suo parametro di potenza.

WDMMultiplexer

Questo componente *Click!* emula un dispositivo ottico in grado di mutiplexare mediante tecnica WDM diverse lunghezze d'onda su un unica fibra. A livello di potenza l'elemento è impostato per attenuare il segnale secondo la formula ideale:

$$attenuazione = 1,2log_2W$$

dove W indica il numero di lunghezze d'onda componenti la fibra.

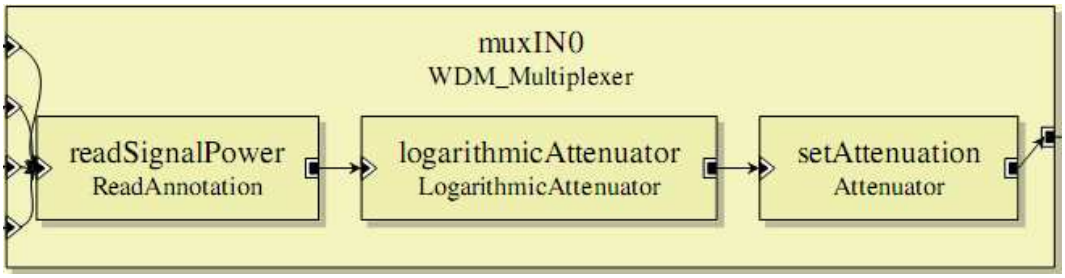

Figura 3.17: WDM Multiplexer

Combiner

Questo componente, complementare allo Splitter, emula un dispositivo ottico in grado di unire un numero variabile di ingressi su un unica uscita. Questa riduzione introduce però una attenuazione del segnale proporzionale al numero di porte in ingresso di un valore logaritmico. A livello di potenza dunque l'elemento è impostato per attenuare il segnale secondo la formula ideale:

$$attenuazione = 10log_{10}N$$

dove N indica il numero di fibre su cui il segnale è replicato.

3.7.6 EOConverter

Questo elemento emula la conversione del nuovo header dal dominio elettronico di controllo al dominio ottico del piano dati.

3.7.7 NewHeader o NH

Qui viene effettuata l'operazione di giunzione tra il payload che ha attravesato la matrice di commutazione e il nuovo header, riconvertito nel dominio ottico. Il nuovo pacchetto ottico è quindi pronto ad uscire dal nodo.

3.7.8 Output Fiber (OF)

Questo elemento modella la fibra di uscita. Al suo interno è presente un contatore in grado di fornire i risultati statistici in fase di valutazione della architettura.

3.8 Comportamento del modello

3.8.1 Analisi dei punti di contesa

La matrice così strutturata presenta due punti di contesa, ovvero dove pacchetti che viaggiano sulla stessa lunghezza d'onda contendono per la stessa fibra di uscita allo stesso istante. Il primo punto è in uscita dai convertitori TWC dedicati alla stessa fibra in ingresso. Infatti due pacchetti che provengono dalla stessa fibra in ingresso non possono essere convertiti alla stessa lunghezza d'onda in uscita dai TWC anche se sono diretti a fibre in uscita differenti. Questo è dovuto allo stadio di multiplexing/combiner a valle dei convertitori. Il secondo punto di contesa è sulle fibre di uscita dove la stessa lunghezza d'onda non può essere usata più di una volta nella stessa fibra. Quindi un pacchetto può essere inoltrato su una lunghezza d'onda W se e solo se la lunghezza d'onda è disponibile in uscita al gruppo di TWC della fibra di input del pacchetto e sulla sua fibra di output. Queste condizioni saranno tenute in conto in fase di progettazione degli algoritmi software di forwarding del nostro modello. Il FM quindi manipola e gestisce la matrice tenendo conto di queste contese.

3.8.2 Analisi del traffico attraverso il nodo

Osserviamo ora in dettaglio come si comporta il traffico quando attraversa il nodo, cercando di dare una visione d'insieme delle tempistiche e dei ruoli dei

componenti presentati finora. Per semplicità analizziamo cosa accade all'interno del nodo ad ogni singolo *timeslot*, durante il quale si possono presentare al massimo W pacchetti in ingresso per ogni fibra, per un totale di M*W (ovvero il numero di fibre ottiche in ingresso al nodo moltiplicato per il numero lunghezze d'onda di ognuna). Nel nostro caso abbiamo dimensionato il nodo in modo da avere M = 2 fibre di ingresso, su ognuna delle quali sono disponibili W = 4 lunghezze d'onda per un massimo totale di M*W = 8 possibili possibili pacchetti presenti in ingresso al nodo per ogni timeslot. Concentriamoci ora sul primo di questi pacchetti, che giunge in ingresso al modello, in modo da descrivere i principali passi svolti dal modello realizzato. Innanzitutto, quando il traffico viene preso in gestione dal modello realizzato mediante *Click!* non viaggia direttamente all'interno della configurazione. Infatti al suo posto viene creato un pacchetto *Click!* formato da un puntatore al pacchetto dati reale in ingresso, da una serie di annotazioni e da altri dati utili al processamento. E' proprio questo a viaggiare all'interno della configurazione e a subire il processamento secondo le operazioni previste dal modello.

Ricevendo dunque i dati da una socket UDP, propria per ogni fibra,ed in particolare per ogni lunghezza d'onda, si effettua la creazione di un pacchetto *Click!* che punti al datagramma UDP ricevuto come mostrato in figura 3.19. Il pacchetto *Click!* viene caratterizzato nelle annotazioni dall'elemento composto InputFiber. Questo elemento infatti memorizza nelle annotazioni la fibra di input dalla quale proviene il pacchetto, la sua lunghezza d'onda e la sua dimensione. Successivamente calcola poi un valore casuale di potenza da attribuire al segnale per la caratterizzazione del comporamento in potenza del nodo.

A questo punto il pacchetto *Click!* relativo al datagram packet UDP arriva al componente HeaderTap. Qui i dati ricevuti devono essere separati nelle due componenti di header e payload ottico rispettivamente. Per far questo l'HeaderTap duplica il pacchetto *Click!* che punta ai dati reali, cancellando dalla prima istanza i dati relativi al payload e nella seconda quelli relativi all'-header e alle guard band. Il payload entrerà in ingresso alla SwitchingMatrix, dove in base alla lunghezza d'onda e alla fibra di ingresso di provenienza viene posto in attesa nella Fiber Delay Line (FDL) ad esso associato. L'header invece viene convertito dal dominio ottico al dominio elettronico dall'apposito elemento OEConverter per essere letto e processato dai dispositivi del piano di controllo per il suo instradamento. Mentre il payload attende nella FDL per un tempo consono al processamento elettronico dell'header, questo entra in coda al Control Element (CE) come mostrato in figura 3.20. Se il *Click!* non ha in ingreso dalla fibra un altro pacchetto allora il CE estrae dalla coda degli headers, secondo logica First-In First-Out (FIFO), il primo header da processare.

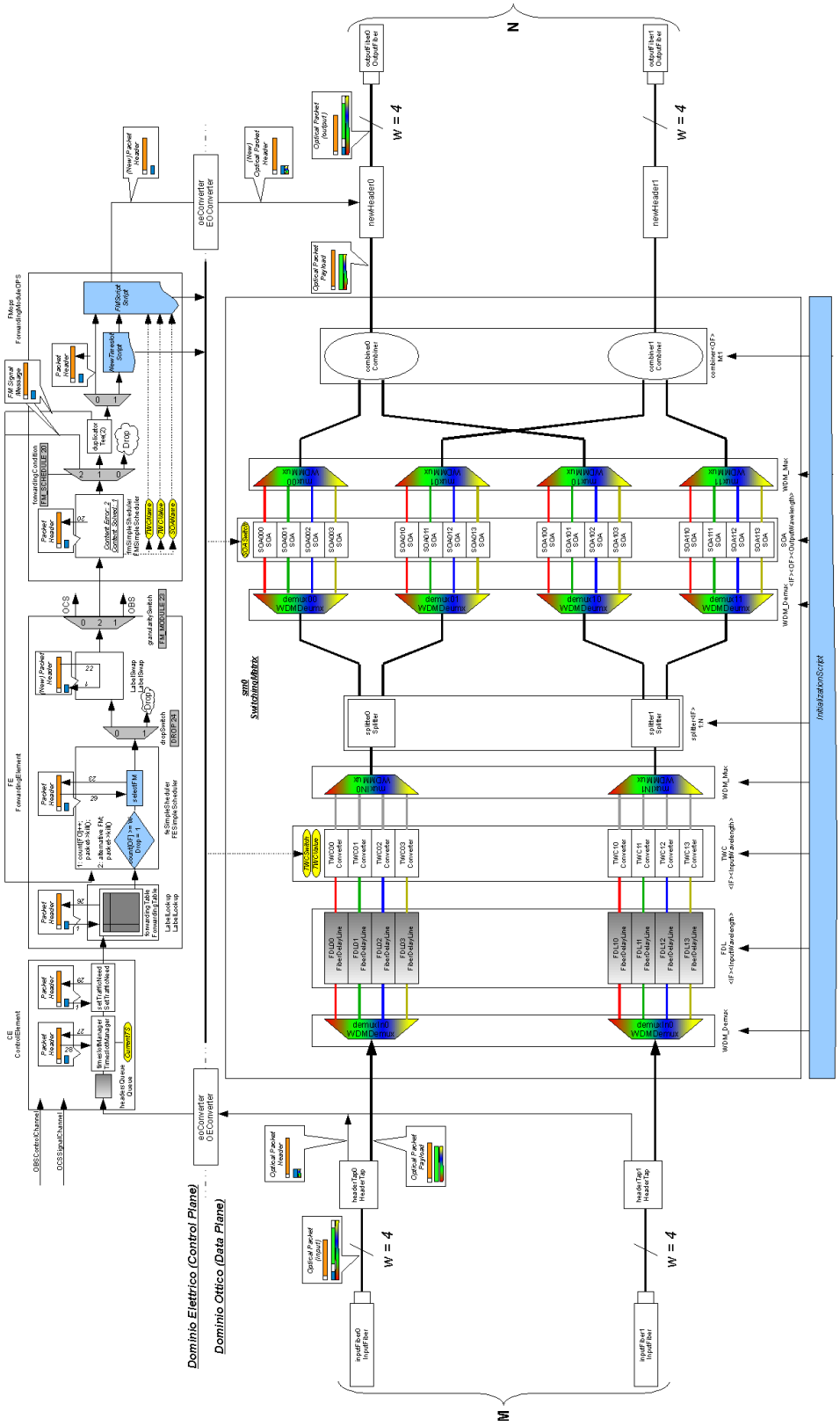

Figura 3.18: Schema generale di funzionamento

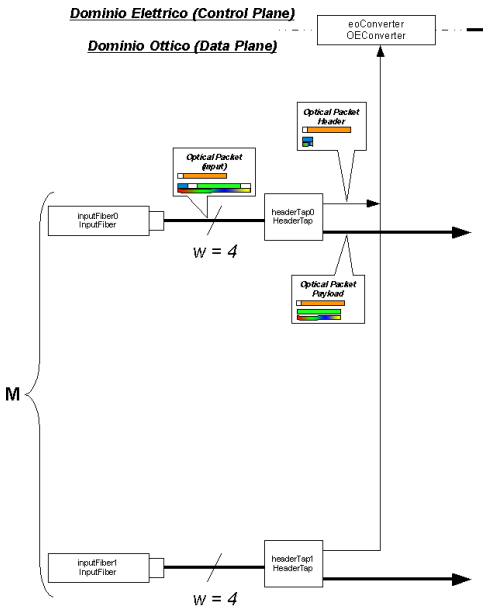

Figura 3.19: Prima fase del processamento dei pacchetti.

Il CE controlla se il pacchetto che sta processando fa parte del timestamp di riferimento o se fa parte di un nuovo timestamp. Se il pacchetto fa parte di un nuovo timestamp allora l'annotazione relativa del pacchetto *Click!* viene impostata ad 1, in modo da segnalare al Forwarding Module (FM) la necessità di resettare la matrice per il nuovo timeslot. A questo punto il CE legge le necessità del traffico in termini di qualità richiesta, lo elabora secondo delle direttive (ad esempio se riconosce una label definita come prioritaria ne accorda le richieste) e imposta il risultato di questa elaborazione come informazione sulla necessità del traffico da passare al Forwarding Element (FE).

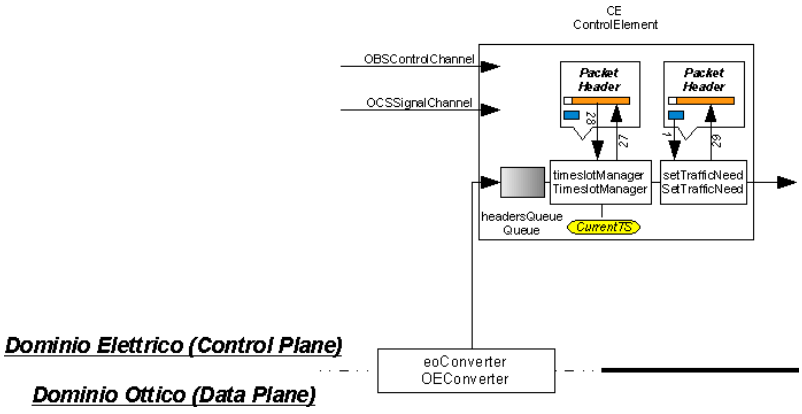

Figura 3.20: Seconda fase del processamento dei pacchetti.

Il pacchetto contenente l'header e pronto al processamento di inoltro viene

dunque passato al componente FE mostrato in figura 3.21. All'interno di questo modulo, vi sono diversi sotto-elementi ognuno con un ruolo particolare. Il primo elemento che il pacchetto incontra è denominato Label Lookup (LL).

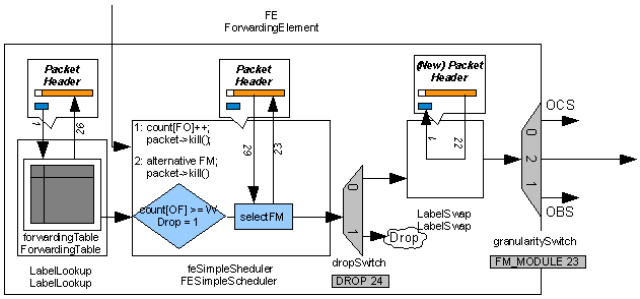

Figura 3.21: Terza fase del processamento dei pacchetti.

In questo elemento viene letta la destination label dall'header ottico nel campo dati del pacchetto *Click!* . Questa label, secondo il modello base del protocollo Generalized Multi-Protocol Label Switching (GMPLS) è confrontata all'interno di una struttura dati interna indicata come Forwarding Table (FT). Questa tabella si compone, in questa prima versione molto semplificata, di tre colonne. La prima colonna è la colonna di ricerca della destination label letta, la seconda indica la nuova destination label e la terza indica quale interfaccia di uscita consente di raggiungerla. Le informazioni della seconda e terza colonna vengono dunque scritte dal LL nelle annotazioni del pacchetto, per un uso futuro. Il secondo componente che l'header attraversa è definito come FESimpleScheduler il quale impementa un semplice algoritmo per lo scheduling dei pacchetti all'interno del nodo mostrato sotto:

```
01: if (schedule == 0) {
02:     if (out counter[n] < W) {
03:         select main FM(need);
04:         send request to FM(n,w,m);}
05:     else {
06:         droppacket/flow;}}
07: if (schedule == 1) {
08:     out counter[n]++;}
09: if schedule==2 {
10:     if (alternative FM == true) {
11:         select alternative FM(need);
12:         send request to FM(n,w,m);}
13:     else {
14:         droppacket/flow;}}
```

A questo punto l'header ottico entra nel terzo ed ultimo componente del FE, il Label Swap (LS). Al suo interno vengono lette le informazioni riguardanti la nuova destination label e l'interfaccia di uscita del nodo ad essa relativa. La nuova destination label viene dunque scritta in quello che diventa da ora il nuovo header ottico del pacchetto. In base alle informazioni inserite dal CE sulle necessità del traffico elaborate poi dall'algoritmo di scheduling del FE si decide quindi a questo punto verso quale FM inoltrare la richiesta per il corrente pacchetto. Come accennato nella corrente implementazione viene fornita soluzione per il traffico di tipo OPS.

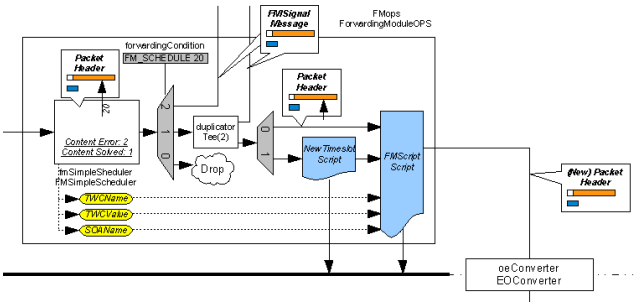

Figura 3.22: Quarta fase del processamento dei pacchetti.

All'interno del Forwarding Module (FM), come mostrato dalla figura 3.22 il nuovo header ottico attraversa il primo elemento denominato FMSimpleScheduler. Tale elemento impementa un semplice pseudo codice per lo scheduling dei pacchetti all'interno del nodo applicando in questo caso una conoscenza esplicita sull'hardware a disposizione per gestire e risolvere eventuali contese nel traffico. In particolare all'interno dello scheduler si esegue lo pseudo codice sottostante, ricordando che a questo punto si ha completa visione dell'hardware della matrice di commutazione e si può effettuare un controllo mirato alla disponibilità dei componenti che la implementano:

```
01: found=0;
02: kinit=0;
03: end_scan=0;
04: while (found == 0 && end_scan == 0) {
05:     if (OF_av[n][k] == 0 && WC_av[m][k] == 0) {
06:         found=1;
07:         OF_av[n][k]=1;
08:         WC_av[m][k]=1;}
09:     else {
10:         k =(k+1)\%W;
11:         if k == kinit[n]{
12:             end_scan=1;}}}
```

```
13: if (found==0) {
14:     schedule=2;
15:     send feedback to FE(schedul);}
16: else {
17:     set devices(n,k,m);
18:     schedule=1;
19:     send feedback to FE(schedul);}
```

Quindi se il pacchetto da inoltrare non genera contese o è possibile risolvere queste allocando il pacchetto su una differente lunghezza d'onda allora è possibile inoltrarlo altrimenti è necessario scartarlo o richiedere un ulteriore soluzione (magari attraverso un FM alternativo mediante l'FE). In entrambi i casi dunque si deve segnalare al FE l'esito dello scheduling da parte del FM e questa operazione è visibile con l'arco di retroazione dal FM al FE che simula la linea di scambio delle informazioni del piano di controllo. Nel caso non riesco ad inoltrare il pacchetto allora segnalo al FE l'esito schedule == 2 altrimenti se riesco nell'operazione segnalo come schedule == 1 e inoltro il nuovo header verso gli elementi di script per l'attuazione dei componenti della matrice per la risoluzione del percorso di inoltro. Il nuovo header, ancora nel FM, viene sottoposto ad un test sul timeslot di appartenenza. In caso l'header appartenga, e quindi sia il primo di un nuovo timeslot, allora si decide di eseguire lo script denominato NewTimeslotScript. Questo elemento permette di ripristinare la matrice nei suoi componenti elementari allo stato iniziale, ovvero con i convertitori e i SOA relativi disabilitati. In questa configurazione dunque il traffico non può raggiungere le uscite se non completamente schedulato. Altrimenti o successivamente a tale passo il nuovo header accede allo script denominato FMScript. Questo è il vero attuatore delle computazioni di scheduling. Attraverso questo componente posso attivare il convertitore necessario, alla lunghezza d'onda voluta e il relativo SOA di destinazione del traffico per ogni header che lo attraversa.

A questo punto come mostrato in figura 3.24 il nuovo header attraversa il convertitore EOConverter che emula la conversione delle informazioni dal dominio elettronico a quello ottico per poi giungere nel componente NewHeader. Qui verrà riunito al payload che attraverserà la matrice allo scadere dell'attesa nella FDL sul piano ottico. Il pacchetto così rigenerato sarà portato in uscita verso la output fiber di destinazione secondo l'algoritmo di scheduling. Infine tutti questi passi si ripetono per ogni nuovo pacchetto in ingresso al nodo e per ogni timeslot elaborato dalla rete.

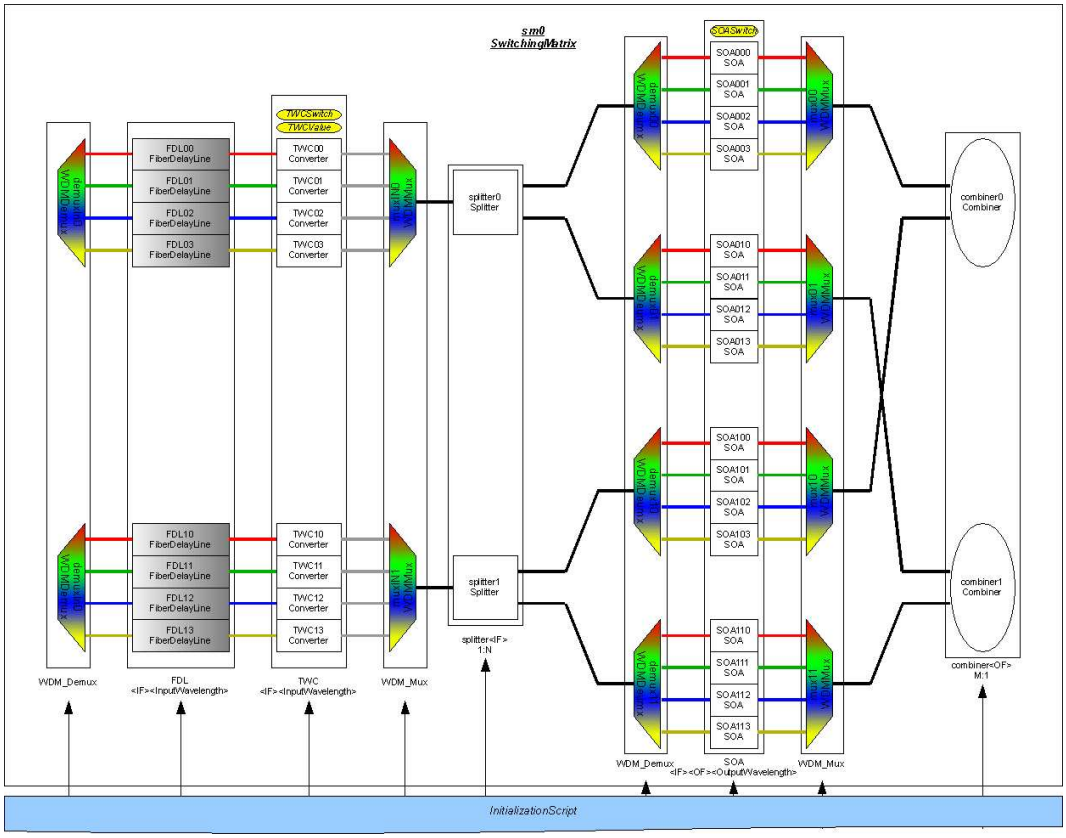

Figura 3.23: La matrice di commutazione in dettaglio

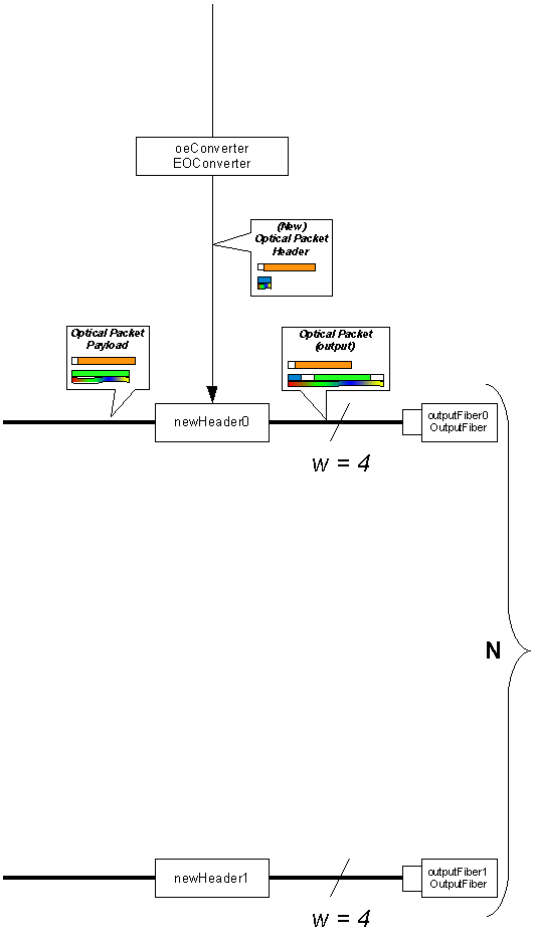

Figura 3.24: Ultima fase del processamento dei pacchetti.

Capitolo 4

Test e valutazioni sul modello

Il test di un programma può essere usato per mostrare la presenza di bug, ma mai per mostrare la loro assenza.

Edsger Wybe Dijkstra,
informatico olandese.
(1930 - 2002)

In questo capitolo esporremo in breve il lavoro di configurazione di una piattaforma hardware dedicata, allestita nel laboratorio del dipartimento. Inoltre forniremo i dati sperimentali dell'esecuzione del modello di router ottico per confrontarne i risultati con dei valori di riferimento provenienti da configurazioni analoghe, testate all'interno di un simulatore.

4.1 La piattaforma hardware di test

Al fine di valutare le prestazioni del modello si è deciso di allestire una piccola configurazione hardware in grado di operare in modo dedicato come emulatore di nodo ottico. A tale scopo sono state approntate in laboratorio tre macchine, configurate per eseguire ognuna la configurazione scelta. In futuro queste macchine saranno inteconnesse secondo uno schema il quale prevede che due di questi calcolatori siano dedicati alla generazione del traffico in ingresso al nodo (mediante una distribuzione bernoulliana di probabilità) mentre sulla restante macchina verrà posta in esecuzione una specifica configurazione del router ottico al fine di raccoglierne i risultati. Un resoconto dettagliato della fase di installazione e *set-up* della architettura allestita in laboratorio è presente in appendice B.3.1.

4.1.1 La distribuzione del traffico

Per fornire dati attendibili, valutare l'architettura proposta e il modello implementato occorre fornire in ingresso un traffico dati simile, o quantomeno comparabile, al traffico reale che viene generato in una rete di questo tipo. Per raggiungere tale scopo, la generazione dei pacchetti viene basata su una distribuzione bernulliana di probabilità. Secondo questa funzione, una volta impostato un valore q, che indica la probabilità di generare con successo un pacchetto, durante la fase di creazione questo viene effettivamente generato se, preso come riferimento un numero pseudo-casuale calcolato, questo è minore della probabilità q, altrimenti, quindi con probabilità 1 - q il pacchetto non viene generato come mostrato in figura 4.1.

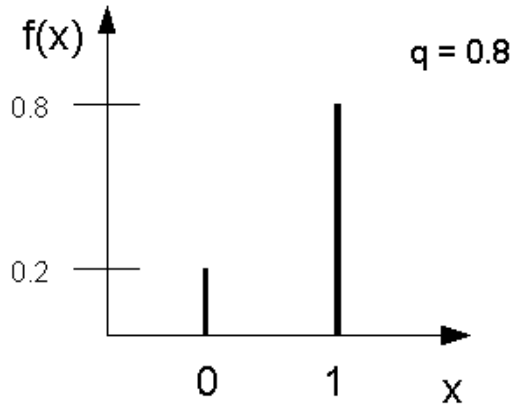

Figura 4.1: Distribuzione bernoulliana con q = 0.8

Come generatore di traffico si è ricorso alla scrittura di un programma in linguaggio C (bernoulligen.c) appositamente creato per la valutazione delle prestazioni. Questo programma è in grado di generare pacchetti secondo la distribuzione bernulliana sopra citata, ed è in grado di ricevere da linea di comando una serie di parametri variabili per la generazione del traffico. A livello funzionale il programma definisce una connessione di rete mediante una *datagram socket* UDP e genera su di questa pacchetti in accordo al valore di probabilità impostato. Il programma consente inoltre di passare una serie di parametri utili ai fini della caratterizzazione del traffico generato. Ad esempio oltre ai dati per la creazione della socket, ovvero l'indirizzo IP e la porta UDP, e il valore della distribuzione di probabilità q, è possibile impostare il seme (*seed*) per la generazione dei valori pseudo-casuali, la durata in millisecondi del timeslot, il numero di timeslot da generare, la dimensione del payload e dell'header dei pacchetti creati e il numero di labels di destinazione da generare.

4.2 Le prestazioni misurate

Le misurazioni effettuate nei vari test mirano a vagliare la probabilità di perdita dei pacchetti, in relazione ad una determinata configurazione del nodo (interfacce di ingresso, di uscita, numero di lunghezze d'onda per canale, ...) e per un determinato fattore di carico relativo al traffico in ingresso oltre ad una misurazione dei tempi di processamento e *scheduling* in modo da determinare le tempistiche di emulazione della architettura.

4.2.1 Packet Loss Probability (PLP)

Questo valore indica la probabilità di perdita dei pacchetti, espressa come risultato della differenza tra i pacchetti in ingresso e quelli effettivamente inoltrati dal nodo sul totale dei pacchetti in ingresso. Questo rapporto è espresso dalla formula:

$$PLP = \frac{pacchetti\_entrati - pacchetti\_usciti}{pacchetti\_entrati}$$

Ovviamente, minore è questo valore migliori sono le prestazioni del nodo (e quindi della sua architettura e del suo algoritmo di scheduling, rispettivamente). Variazioni del numero di ingressi, uscite, numero di lunghezze d'onda del canale, così come delle possibili implementazioni di algoritmi di scheduling differenti sono tutti fattori che influenzano il valore espresso dalla PLP. Scopo dei test sarà esporre tali differenze al variare dei suddetti fattori per evidenziare quali soluzioni architetturali, in termini di prestazioni, risultano essere più efficaci.

4.2.2 Tempo di processamento elettronico

Uno dei limiti in cui ci si è imbattuti nella realizzazione del modello sono stati i tempi di emulazione del singolo timeslot. Queste tempistiche sono risultate essere molto alte e non comparabili ai livelli di una equivalente simulazione. I fattori di questa limitazione sono stati individuati e aprono la strada a future e migliori implementazioni. Forniamo ora una serie di fattori che secondo l'autore influenzano queste tempistiche e alcune soluzioni applicabili nella futura evoluzione del modello.

Esecuzione in ambiente *userlevel*

Uno dei fattori che influenzano le tempistiche è l'esecuzione della configurazione *Click!* in ambiente *userlevel*. In questa modalità infatti è il sistema operativo

che assegna risorse e tempistiche al *Click!* e, a fronte di richieste da parte di altri programmi, queste vengono ripartite piu o meno equamente. Nelle prove effettuate si è tenuto conto di questo dedicando il sistema all'esecuzione della configurazione *Click!* , disabilitando l'interfaccia grafica e i servizi superflui, ma una esecuzione a livello kernel comporterebbe sicuramente migliori prestazioni, riducendo la tempistica di esecuzione del singolo timeslot.

Scrittura a video e su file

Un altro dei fattori rilevanti è l'output a video e su file dei risultati. L'accesso al video così come alla RAM o al disco rallentano notevolmente l'esecuzione. Si è così ridotto al minimo la produzione di messagi di testo, ma una soluzione di immagazzinamento delle informazioni all'interno di variabili di processo, riversate su file o disco alla fine della emulazione, incrementerebbe ulteriormente le prestazioni.

Esecuzione degli script FM

Questo è forse il fattore macroscopico delle tempistiche del processamento elettronico degli *header*. Per impostare e resettare i dispositivi di commutazione (ad esempio WC, SOA, ...) della SM si è ricorsi all'uso di codice *script* in linguaggio *Click!* . Se pur comodi come strumento, questi script usano codice a livello applicativo, obbligando il processo a risalire fino all'interpretazione dello stesso per poi tornare ad eseguire codice compilato. Empiricamente si è trovato che mentre l'attraversamento di un pacchetto attraverso un semplice componente *Click!* , composto dal solo codice C++, è nell'ordine dei 0,0001 secondi, lo stesso attraversamento del componente FM contenente gli script di attuazione e reset, impiega un ordine di grandezza superiore facendo aumentare così notevolmente il tempo di emulazione del processamento elettronco. Soluzione proposta, anche tra gli sviluppi futuri, è quella di implementare tali script direttamente nel codice C++ del componente FM.

Sequenzializzazione del *Click!*

Ultimo vincolo alle tempistiche di processamento è la sequenzializzazione imposta ai pacchetti in ingresso al router. Prima di poter processare un pacchetto infatti occorre che il precedente sia già stato elaborato o inserito in attesa in una coda. Una possibile soluzione è emersa recentemente nella mailing list del *Click!* dove si prospettava la possibilità di implementazione a livello multithread del motore di elaborazione delle configurazioni.

4.3 Test e risultati

4.3.1 Lo script di lancio dei test

Nei test di valutazione, l'esecuzione della configurazione *Click!* relativa, e del programma generatore di traffico è stata affidata ad un particolare script *bash*. Questo file (identificato dalla dicitura bernoulligenerator_real_traffic.sh) permette di mandare in esecuzione una particolare configurazione *Click!* e, sucessivamente, creare tante istanze del programma di generazione dei pacchetti quante sono le interfacce di ingresso del nodo, ed in particolare quante sono le lunghezze d'onda per ognuna di esse. Inoltre è possibile impostare mediante lo script un determinato range di valori di carico su cui reiterare le simulazioni. In questo modo, l'output generato dalla configurazione *Click!* posta in ascolto, verrà rediretto su dei file di testo (output_NxN_w_q.txt) distinti per ogni configurazione e per ogni valore di carico. Analizzando questi file è possibile estrapolare le informazioni rigurdanti i contatori dei pacchetti generati, scartati (sia dal FE che dal FM) e quindi la loro differenza ai fini di poter calcolare la PLP e le prestazioni della configurazione per un determinato fattore di carico.

4.3.2 Test I

Per il primo test di valutazione del modello si è definita una configurazione *Click!* con M = 2 fibre in ingresso e N = 2 fibre in uscita, ognuna con a disposizione W = 4 lunghezze d'onda. Generando quindi un numero fissato di *timeslot* con valori di carico q variabili per un traffico a distribuzione bernoulliana, siamo riusciti ad ottenere vari riferimenti in termini di pacchetti inoltrati o persi a fronte dei diversi valori di carico. Inoltre è stato possibile intercettare i punti di perdita all'interno dell'algoritmo di scheduling fornendo un importante misura per le prestazioni dello stesso. Infatti vi può essere perdita in base alla non disponibilità di interfacce di uscita (controllo ad opera del FE) o di lunghezze d'onda (controllo ad opera del FM) il che varia la probabilità che si verifichino delle contese, risolvibili o meno da parte dello *scheduler*. Mostriamo ora dapprima in forma tabellare e successivamente graficati i risultati ottenuti mediante simulazione prima e l'emulazione del modello implementato poi.

I risultati della simulazione

Al fine di avere un termine di paragone che validasse la correttezza del modello emulato si è deciso di produrre alcune simulazioni in modo da avere un riscontro

in termini di probabilità di perdita dei pacchetti rispetto ad un determinato carico.

Test I - Simulazione:	
Input fibers:	2
Output fibers:	2
Wavelengths per fiber:	4
Emulated number of packets:	100.000
Packet Loss Probability (q=0.9):	0,1022240000
Packet Loss Probability (q=0.8):	0,0725626000
Packet Loss Probability (q=0.7):	0,0483598000
Packet Loss Probability (q=0.6):	0,0294509000
Packet Loss Probability (q=0.5):	0,0160509000
Packet Loss Probability (q=0.4):	0,0071689900
Packet Loss Probability (q=0.3):	0,0025680000
Packet Loss Probability (q=0.2):	0,0005529990

Tabella 4.1: Risultati Test I - Simulazione

Come si evince dalla tabella 4.1 sono riportati i valori di PLP per determinati fattori di carico sulla generazione totale dei pacchetti. Ci aspettiamo dunque che la configurazione emulata restituisca dei risultati concordi con la simulazione al fine di valutarne l'effettiva funzionalità, a fronte di un più complesso e dettagliato meccanismo di implementazione del modello.

I risultati dell'emulazione

tab:RisultatiTestIIbSimulazione

Test I - Emulazione:	
Input fibers:	2
Output fibers:	2
Wavelengths per fiber:	4
Emulated number of timeslot:	16.000
Packet Loss Probability (q=0.9):	0,1017533321
Packet Loss Probability (q=0.8):	0,0713337628
Packet Loss Probability (q=0.7):	0,0490464733
Packet Loss Probability (q=0.6):	0,0294824964
Packet Loss Probability (q=0.5):	0,0172571944
Packet Loss Probability (q=0.4):	0,0077419857
Packet Loss Probability (q=0.3):	0,0024322628
Packet Loss Probability (q=0.2):	0,0006269838

Tabella 4.2: Risultati Test I - Emulazione

Come ci aspettavamo, i risultati forniti dalla emulazione sono comparabili sia in termini di ordine di grandezza sia in valore con i risultati della simulazione. Per una migliore chiarezza vediamo ora di graficare questi dati per confrontarli e far emergere i risultati attesi.

Confronto

Sovrapponendo i grafici ottenuti mediante interpolazione dei dati della emulazione e della simulazione si ottiene il grafico mostrato in figura 4.2.

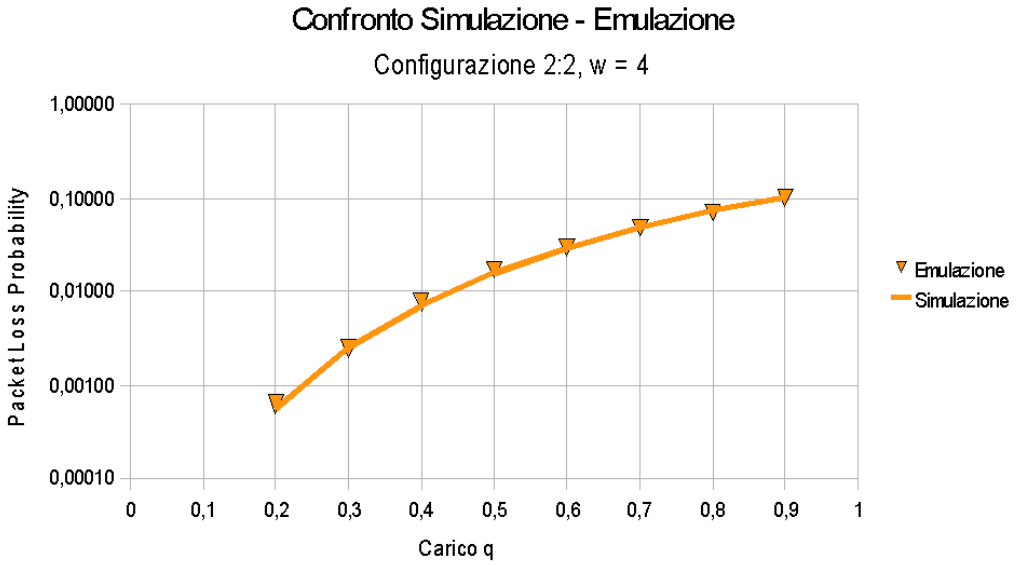

Figura 4.2: Confronto delle curve di simulazione ed emulazione per il TestI

I due modelli coincidono in termini di prestazioni e PLP per i diversi fattori di carico espressi, come atteso. I valori della simulazioni sono graficati sotto forma di linea continua mentre i valori della emulazione sono presenti sotto forma di punti. Alcuni discostamenti, soprattutto nella parte sinistra del grafico sono dovuti al fatto che, tenendo fisso il numero di timeslot, si produce un quantitativo minore di pacchetti per valori di carico più bassi e quindi il risultato tende a non essere preciso in quanto calcolato su un minor numero di pacchetti. Una volta comprovata la sostanziale eguaglianza dei risultati e la funzionalità del modello la differenza sostanziale rimane nel fatto che, mentre nella simulazione ci limitiamo a generare via codice gli algoritmi di scheduling di una architettura, nella emulazione riproduciamo l'intero processamento del traffico nel nodo, interpretando ogni singolo componente, realizzando così un modello più complesso e dettagliato della tecnologia reale.

4.3.3 Test IIa

Per il secondo test di valutazione del modello, si è definita una configurazione *Click!* con M = 4 fibre in ingresso e N = 4 fibre in uscita ognuna con a disposizione, in questa prima variante del TestII W = 4 lunghezze d'onda. Generando quindi un numero fissato di *timeslot* con valori di carico q variabili per un traffico a distribuzione bernoulliana, siamo riusciti ad ottenere vari riferimenti in termini di pacchetti inoltrati e pacchetti persi a fronte di diversi valori di carico. Inoltre è stato possibile intercettare i punti di perdita all'interno dell'algoritmo di scheduling fornendo un importante misura per le prestazioni dello stesso. Infatti vi può essere perdita in base alla non disponibilità di interfacce di uscita (controllo ad opera del FE) o di lunghezze d'onda (controllo ad opera del FM) il che varia la probabilità che si verifichino delle contese, risolvibili o meno da parte dello scheduler. Mostriamo ora dapprima in forma tabellare e successivamente graficati i risultati ottenuti mediante simulazione prima ed emulazione del modello implementato poi.

I risultati della simulazione

Al fine di avere un termine di paragone che validasse la correttezza del modello sviluppato si è deciso di produrre alcune simulazioni in modo da avere un riscontro in termini di probabilità di perdita di pacchetti rispetto ad un determinato carico.

Test IIa - Simulazione:	
Input fibers:	4
Output fibers:	4
Wavelengths per fiber:	4
Emulated number of packets:	100.000
Packet Loss Probability (q=0.9):	0,1477520000
Packet Loss Probability (q=0.8):	0,1134840000
Packet Loss Probability (q=0.7):	0,0818013000
Packet Loss Probability (q=0.6):	0,0537525000
Packet Loss Probability (q=0.5):	0,0311562000
Packet Loss Probability (q=0.4):	0,0150742000
Packet Loss Probability (q=0.3):	0,0055756000
Packet Loss Probability (q=0.2):	0,0012594000

Tabella 4.3: Risultati Test IIa - Simulazione

Come si evince dalla tabella 4.3 sono riportati i valori di PLP per determinati fattori di carico sulla generazione totale dei pacchetti. Ci aspettiamo dunque che la configurazione emulata restituisca dei risultati concordi con

la simulazione al fine di valutarne l'effettiva funzionalità, a fronte di un più complesso e dettagliato meccanismo di implementazione del modello.

I risultati dell'emulazione

I risultati analoghi per l'emulazione sono visibili nella tabella 4.4.

Test IIa - Emulazione:	
Input fibers:	4
Output fibers:	4
Wavelengths per fiber:	4
Emulated number of timeslot:	8.000
Packet Loss Probability (q=0.9):	0,1512347820
Packet Loss Probability (q=0.8):	0,1157344800
Packet Loss Probability (q=0.7):	0,0822657904
Packet Loss Probability (q=0.6):	0,0550687386
Packet Loss Probability (q=0.5):	0,0311675333
Packet Loss Probability (q=0.4):	0,0138047859
Packet Loss Probability (q=0.3):	0,0055533930
Packet Loss Probability (q=0.2):	0,0012114107

Tabella 4.4: Risultati Test IIa - Emulazione

Come ci aspettavamo, i risultati forniti dalla emulazione sono comparabili sia in termini di ordine di grandezza sia in valore con i risultati della simulazione. Per una migliore chiarezza vediamo ora di graficare questi dati per confrontarli e far emergere i risultati attesi.

4.3.4 Test IIb

Per il secondo test di valutazione del modello, si è definita una configurazione *Click!* con M = 4 fibre in ingresso e N = 4 fibre in uscita ognuna con a disposizione, in questa seconda variante del TestII W = 8 lunghezze d'onda. Generando quindi un numero fissato di *timeslot* con valori di carico q variabili, per un traffico a distribuzione bernoulliana siamo riusciti ad ottenere vari riferimenti in termini di pacchetti inoltrati e pacchetti persi a fronte di diversi valori di carico. Inoltre è stato possibile intercettare i punti di perdita all'interno dell'algoritmo di scheduling fornendo un importante misura per le prestazioni di tale algoritmo. Infatti vi può essere perdita in base alla non disponibilità di interfacce di uscita (controllo ad opera del FE) o di lunghezze d'onda (controllo ad opera del FM) il che varia la probabilità che si verifichino delle contese, risolvibili o meno da parte dello scheduler. Mostriamo ora dapprima in forma

tabellare e successivamente graficati i risultati ottenuti mediante simulazione prima ed emulazione del modello implementato poi.

I risultati della simulazione

Al fine di avere un termine di paragone che validasse la correttezza del modello sviluppato si è deciso di produrre alcune simulazioni in modo da avere un riscontro in termini di probabilità di perdita di pacchetti rispetto ad un determinato carico.

Test IIb - Simulazione:	
Input fibers:	4
Output fibers:	4
Wavelengths per fiber:	8
Emulated number of packets:	100.000
Packet Loss Probability (q=0.9):	0,0934308000
Packet Loss Probability (q=0.8):	0,0614045000
Packet Loss Probability (q=0.7):	0,0353349000
Packet Loss Probability (q=0.6):	0,0164120000
Packet Loss Probability (q=0.5):	0,0059629300
Packet Loss Probability (q=0.4):	0,0015869800
Packet Loss Probability (q=0.3):	0,0002329970

Tabella 4.5: Risultati Test IIb - Simulazione

Come si evince dalla tabella 4.5 sono riportati i valori di PLP per determinati fattori di carico sulla generazione totale dei pacchetti. Ci aspettiamo dunque che la configurazione emulata restituisca dei risultati concordi con la simulazione al fine di valutarne l'effettiva funzionalità, a fronte di un più complesso e dettagliato meccanismo di implementazione del modello.

I risultati dell'emulazione

I risultati analoghi per l'emulazione sono visibili nella tabella 4.6.

Come ci aspettavamo, i risultati forniti dalla emulazione sono comparabili sia in termini di ordine di grandezza sia in valore con i risultati della simulazione. Per una migliore chiarezza vediamo ora di graficare questi dati per confrontarli e far emergere i risultati attesi.

Confronto

Per la prima variante, sovrapponendo i grafici ottenuti mediante interpolazione dei dati della emulazione e della simulazione si ottiene il grafico mostrato in figura 4.3.

Test IIb - Emulazione:	
Input fibers:	4
Output fibers:	4
Wavelengths per fiber:	8
Emulated number of timeslot:	8.000
Packet Loss Probability (q=0.9):	0,0959929164
Packet Loss Probability (q=0.8):	0,0604783644
Packet Loss Probability (q=0.7):	0,0321538393
Packet Loss Probability (q=0.6):	0,0145970235
Packet Loss Probability (q=0.5):	0,0051573699
Packet Loss Probability (q=0.4):	0,0013272954
Packet Loss Probability (q=0.3):	0,0003015507

Tabella 4.6: Risultati Test IIb - Emulazione

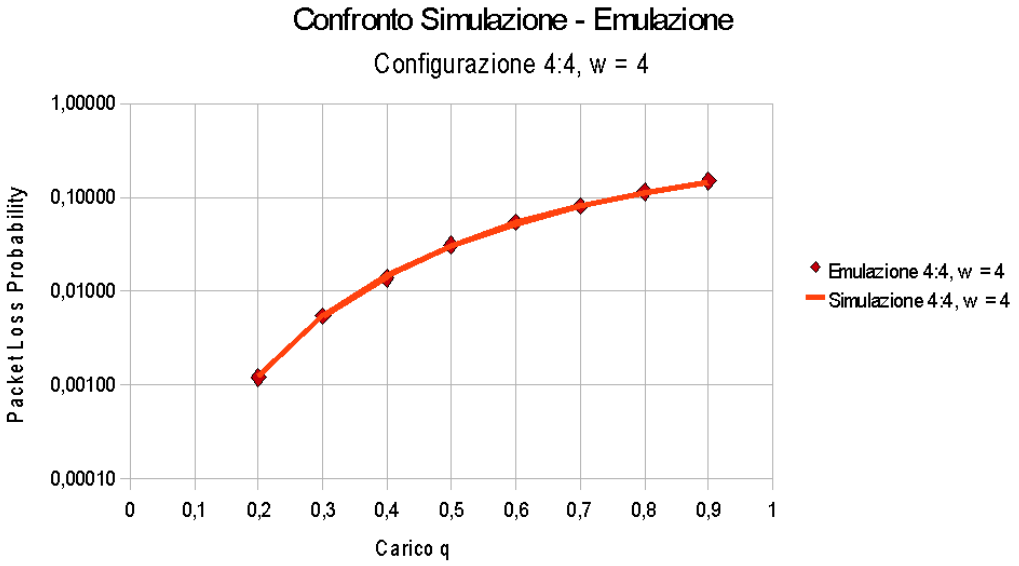

Figura 4.3: Confronto delle curve di simulazione ed emulazione per il TestIIa

Come si evince dal grafico, i due modelli coincidono in termini di prestazioni e PLP per i diversi fattori di carico espressi, come atteso. I valori della simulazioni sono estressi sotto forma di linea continua mentre i valori della emulazione sono espressi sotto forma di punti. Alcuni discostamenti, soprattutto nella parte sono dovuti al fatto che tenendo fisso il numero di timeslot si produce un quantitativo minore di pacchetti per valori di carico più bassi e quindi il risultato tende a non essere preciso in quanto calcolato su un minor numero di pacchetti.

Per la seconda variante, anche qui sovrapponendo i grafici ottenuti mediante interpolazione dei dati della emulazione e della simulazione si ottiene il grafico mostrato in figura 4.4.

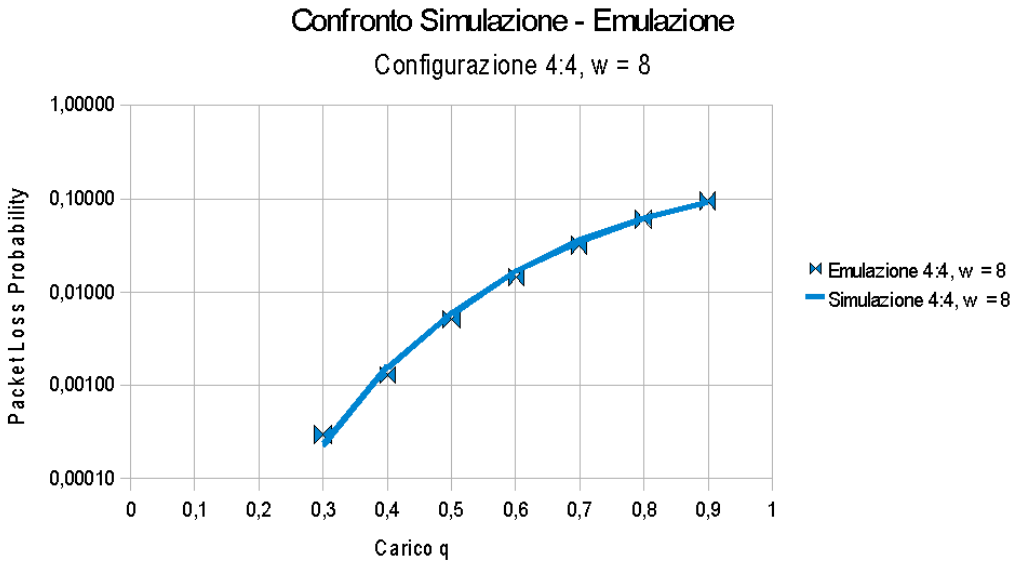

Figura 4.4: Confronto delle curve di simulazione ed emulazione per il TestIIb

Come si evince dal grafico, i due modelli coincidono in termini di prestazioni e PLP per i diversi fattori di carico espressi, come atteso. I valori della simulazioni sono estressi sotto forma di linea continua mentre i valori della emulazione sono espressi sotto forma di punti. Alcuni discostamenti, soprattutto nella parte sono dovuti al fatto che tenendo fisso il numero di timeslot si produce un quantitativo minore di pacchetti per valori di carico più bassi e quindi il risultato tende a non essere preciso in quanto calcolato su un minor numero di pacchetti. Come già detto la differenza sostanziale rimane nel fatto che mentre nella simulazione ci limitiamo a generare via codice gli algoritmi di scheduling di una architettura nella emulazione riproduciamo l'intero processamento del traffico nel nodo, emulando ogni singolo componente, realizzando così un modello più complesso e dettagliato della tecnologia reale.

Andiamo ora a sovrapporre le curve ottenute per il TestII al variare del numeo di lunghezze d'onda disponibili, W = 4 nel caso del TestIIa e W = 8 nel caso del TestIIb. I risultati graficati sono visibili in figura 4.5.

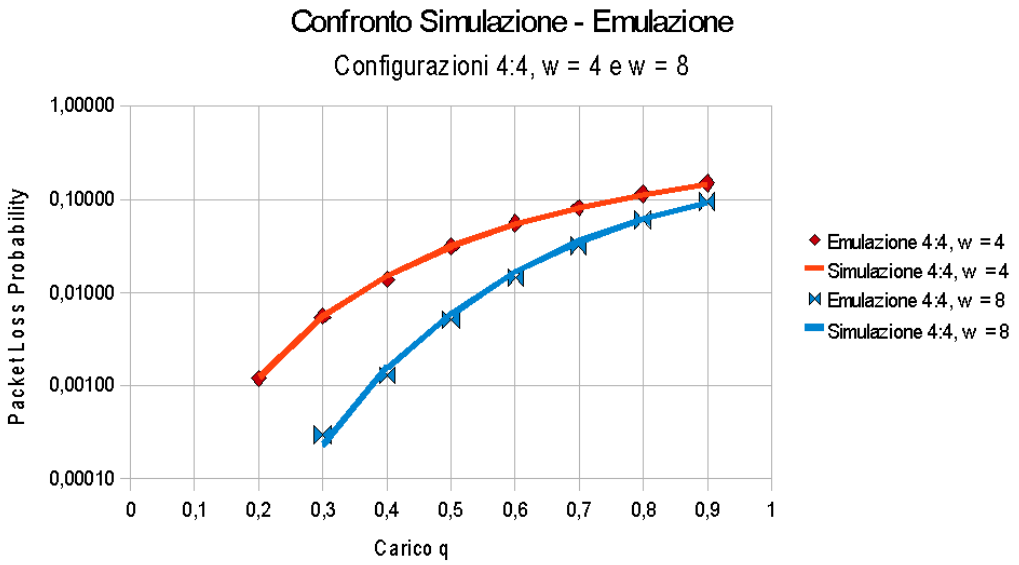

Figura 4.5: Confronto delle curve di simulazione ed emulazione dei TestIIa e b

Come emerge dal grafico all'aumentare del numero di lunghezze d'onda disponibili per canale, diminuisce la probabilità di perdita, soprattutto ai bassi carichi come evidenziato dall'allontanamento delle due curve per valori prossimi allo zero del carico. Questo comportamento è corretto in quanto con un maggior numero W di lunghezze d'onda disponibili è possibile trovare soluzione a più situazioni di contese rispetto ad un valore di W inferiore, da qui risultati di probabilità di perdita più bassi.

4.3.5 Test III

Per il terzo ed ultimo test di valutazione del modello, si è definita una configurazione *Click!* con M = 8 fibre in ingresso e N = 8 fibre in uscita ognuna con a disposizione W = 4 lunghezze d'onda. Generando quindi un numero fissato di *timeslot* con valori di carico q variabili per un traffico a distribuzione bernoulliana, siamo riusciti ad ottenere vari riferimenti in termini di pacchetti inoltrati e pacchetti persi a fronte di diversi valori di carico. Inoltre è stato possibile intercettare i punti di perdita all'interno dell'algoritmo di scheduling fornendo un importante misura per le prestazioni dello stesso. Infatti vi può essere perdita in base alla non disponibilità di interfacce di uscita (controllo ad opera del FE) o di lunghezze d'onda (controllo ad opera del FM) il che varia

la probabilità che si verifichino delle contese, risolvibili o meno da parte dello scheduler. Mostriamo ora dapprima in forma tabellare e successivamente graficati i risultati ottenuti mediante simulazione prima ed emulazione del modello implementato poi.

I risultati della simulazione

Al fine di avere un termine di paragone che validasse la correttezza del modello sviluppato si è deciso di produrre alcune simulazioni in modo da avere un riscontro in termini di probabilità di perdita di pacchetti rispetto ad un determinato carico.

Test III - Simulazione:	
Input fibers:	8
Output fibers:	8
Wavelengths per fiber:	4
Emulated number of packets:	100.000
Packet Loss Probability (q=0.9):	0,1677380000
Packet Loss Probability (q=0.8):	0,1310410000
Packet Loss Probability (q=0.7):	0,0964280000
Packet Loss Probability (q=0.6):	0,0654390000
Packet Loss Probability (q=0.5):	0,0385870000
Packet Loss Probability (q=0.4):	0,0191339000
Packet Loss Probability (q=0.3):	0,0072959900
Packet Loss Probability (q=0.2):	0,0016550000

Tabella 4.7: Risultati Test III - Simulazione

Come si evince dalla tabella 4.7 sono riportati i valori di PLP per determinati fattori di carico sulla generazione totale dei pacchetti. Ci aspettiamo dunque che la configurazione emulata restituisca dei risultati concordi con la simulazione al fine di valutarne la effettiva funzionalità, a fronte di più complesso e dettagliato meccanismo di implementazione del modello.

I risultati dell'emulazione

I risultati analoghi per l'emulazione sono visibili nella tabella 4.8.

Come ci aspettavamo, i risultati forniti dalla emulazione sono comparabili sia in termini di ordine di grandezza sia in valore con i risultati della simulazione. Per una migliore chiarezza vediamo ora di graficare questi dati per confrontarli e far emergere i risultati attesi.

Test III - Emulazione:	
Input fibers:	8
Output fibers:	8
Wavelengths per fiber:	4
Emulated number of timeslot:	8.000
Packet Loss Probability (q=0.9):	0,1794762744
Packet Loss Probability (q=0.8):	0,1348868606
Packet Loss Probability (q=0.7):	0,0988361703
Packet Loss Probability (q=0.6):	0,0641028986
Packet Loss Probability (q=0.5):	0,0388308387
Packet Loss Probability (q=0.4):	0,0174959746
Packet Loss Probability (q=0.3):	0,0073687523
Packet Loss Probability (q=0.2):	0,0016562167

Tabella 4.8: Risultati Test III - Emulazione

Confronto

Sovrapponendo i grafici ottenuti mediante interpolazione dei dati della emulazione e della simulazione si ottiene il grafico mostrato in figura 4.6.

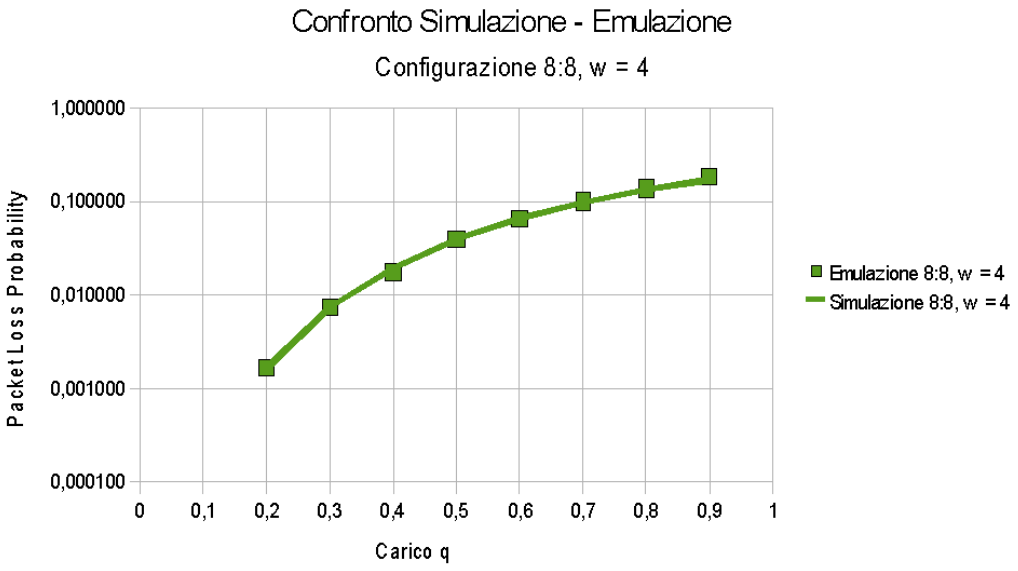

Figura 4.6: Confronto delle curve di simulazione ed emulazione per il TestIII

Come si evince dal grafico, i due modelli coincidono in termini di prestazioni e PLP per i diversi fattori di carico espressi, come atteso. I valori della simulazioni sono estressi sotto forma di linea continua mentre i valori della emulazione sono espressi sotto forma di punti. Alcuni discostamenti, soprattutto

nella parte sono dovuti al fatto che tenendo fisso il numero di timeslot si produce un quantitativo minore di pacchetti per valori di carico più bassi e quindi il risultato tende a non essere preciso in quanto calcolato su un minor numero di pacchetti. Una volta comprovata dalla comparazione dei risultati la funzionalità del modello la differenza sostanziale rimane nel fatto che mentre nella simulazione ci limitiamo a generare via codice gli algoritmi di scheduling di una architettura nella emulazione riproduciamo l'intero processamento del traffico nel nodo, emulando ogni singolo componente, realizzando così un modello più complesso e dettagliato della tecnologia reale.

4.4 Confronto sui test di valutazione

Sovrapponendo ora i grafici ottenuti da tutti i test svolti, mediante l'interpolazione dei dati delle emulazioni e delle simulazioni si ottiene il grafico mostrato in figura 4.7.

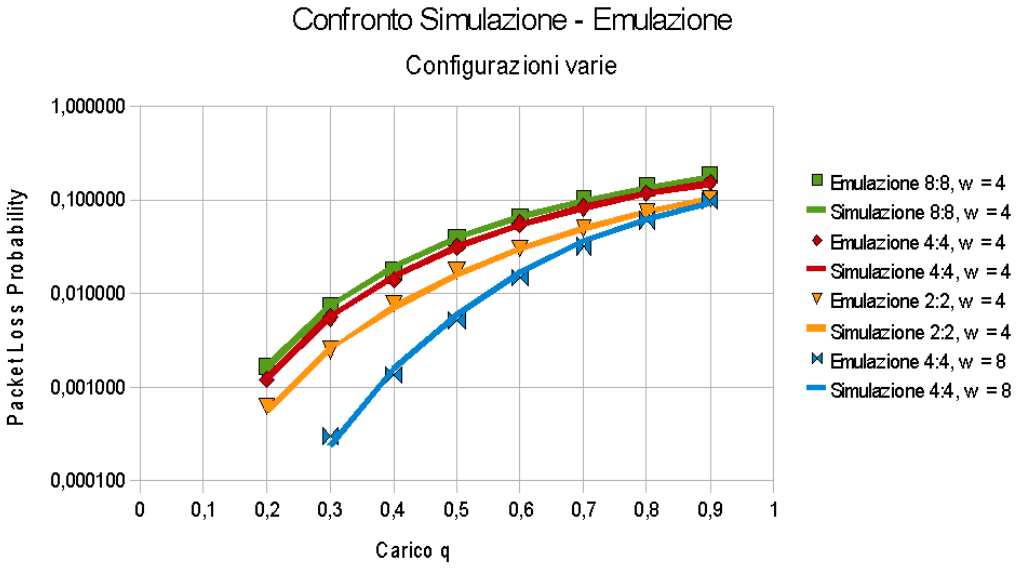

Figura 4.7: Confronto delle curve di simulazione ed emulazione per i test

Come visibile le varie configurazioni presentano prestazioni di perdita differenti dovute alla variazione del numero di ingressi, del numero delle uscite e del numero di lunghezze d'onda disponibili per canale oltre che ovviamente per i valori di carico del traffico generato. Abbiamo già espresso particolare interesse per le curve dei test IIa e IIb in cui viene mostrato come, per lo stesso numero di ingressi e uscite, al variare delle lunghezze d'onda disponibili varia anche la PLP avendo lo scheduler a disposizione più soluzioni per risolvere le

contese. Possiamo quindi mostrare come la probabilità di perdita aumenta al crescere del numero delle interfacce di ingresso presenti nel nodo. Le curve delle emulazioni dei test I, IIa e III evidenziano come la perdita cresca al crescere di M ma al tempo stesso si attesti più o meno asintoticamente sopra un certo valore. Mentre la differenza tra la configurazione con M = 2 e M = 8 è più marcata, questa si riduce tra M = 4 e M = 8 e tenderà a ridursi ancora tra M = 8 ed un ipotetica configurazione M = 16 o M = 32.

Conclusioni

Per aspera sic itur ad astra.

Seneca, filosofo, politico e
drammaturgo romano.
(4 a.C. - 65)

Il completamento del lavoro di tesi ha richiesto 13 settimane di lavoro, suddiviso in documentazione, analisi ed implementazione dei concetti oggetto di ricerca. Osserveremo ora in dettaglio il punto di arrivo del lavoro fin qui svolto e i suoi possibili sviluppi futuri.

4.5 Risultati ottenuti

Il bilancio del lavoro vede circa 4 mesi di applicazione di cui oltre due terzi spesi presso il Dipartimento di Elettronica, Informatica e Sistemistica (DEIS) a stretto contatto con i ricercatori e i laureandi dedicati a questo progetto. L'impementazione realizzata è conforme nelle specifiche alle direttive ForCES e rispecchia i valori di PLP dei modelli più semplici di simulazione. Essa costituisce quindi una piattaforma di lavoro modulare e estremamente estendibile in vari aspetti.

4.6 Sviluppi futuri

4.6.1 Riduzione delle tempistiche di emulazione

Uno dei primi sviluppi da curare sarà probabilmente la riduzione delle tempistiche delle emulazioni. Come espresso nel capitolo dedicato ai test, alcuni fattori sono stati evidenziati e alcune soluzioni proposte.

4.6.2 Implementazione della multi-granularità

Il modello potrà poi essere in futuro esteso nei rimanenti paradigmi OCS e OBS rendendo completa l'architettura nella sua accezione multi-granulare. Questo comporterà la definizione di nuovi elementi di scheduling che prevedano questa possibilità e gestiscano opportunamente le informazioni provenienti dagli elementi di controllo di queste modalità.

4.6.3 Implementazione di un modello reale del consumo di potenza

Un aspetto interessante, in accordo anche con la tendenza del settore in questi ultimi mesi, è quello di poter monitorare i livelli di potenza e consumo all'interno del router. Una prima, seppur elementare, implementazione è già disponibile come visto in precedenza, ma si potrebbe estenderla al fine di emulare concretamente i valori in gioco in un dispositivo reale e restituire informazioni utili sul consumo di potenza al variare delle architetture e degli algoritmi di scheduling.

4.7 Un pò di numeri

- 25.700.000 circa le pagine riguardanti le reti ottiche.
- 355.000 circa le pagine riguardanti OPS.
- 85.400 circa le pagine riguardanti *Click!* .
- 600 ore circa di lavoro su 13 settimane.
- 300 circa le pagine di documentazione varia stampate.
- 200 ed oltre le pagine web visitate, di cui il 95% in lingua inglese.
- 120 circa le pagine scritte per la tesi.
- 80 i *bookmark* collezionati nel corso del progetto.
- 2 i forum frequentemente visitati.
- 3 le macchine usate.

Appendice A

Appendice A: Il software *Click!* Modular Router

La disumanità del computer sta nel fatto che, una volta programmato e messo in funzione, si comporta in maniera perfettamente onesta.

Isaac Asimov, scrittore russo naturalizzato statunitense. (1920 - 1992)

In questa prima appendice presenteremo i concetti base, l'architettura e il linguaggio della piattaforma software utilizzata nello sviluppo del nostro modello.

A.1 Introduzione a *Click!*

Il *Click!* è una architettura software open source modulare, orientata alla realizzazione di una vasta gamma di dispositivi come router, processori di pacchetti, sorgenti di traffico, Ethernet switch, firewall .... basata su piattaforma GNU/Linux e sviluppata dal Massachusetts Institute of Technology (MIT) è ampiamente documentata e distribuita gratuitamente sul sito: `http://www.pdos.lcs.mit.edu/click`.

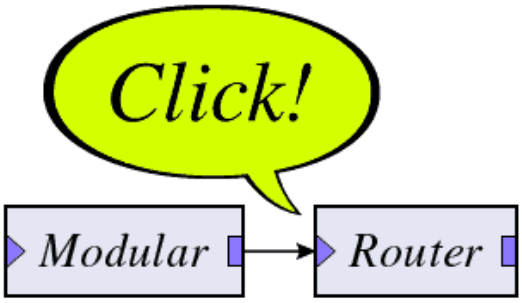

Figura A.1: Logo del software Click Modular Router

A.2 Architettura

Un qualsiasi dispositivo *Click!* è modellato esclusivamente attraverso l'aggregazione di moduli di elaborazione dei pacchetti chiamati *elementi* e non esiste ulteriore astrazione per un componente (del dispositivo) oltre a questa. Gli elementi inoltre sono collegati tra loro mediante delle linee orientate, che rappresentano il flusso dei pacchetti, dette *connessioni*. Ogni elemento implementa semplici funzioni del router come la classificazione, l'accodamento, lo scheduling e l'interfacciamento con i dispositivi di rete. Un insieme di elementi connessi con più linee orientate, le connessioni, rappresenta una *configurazione*, ovvero il modello del dispositivo che vogliamo simulare.

A.2.1 Elementi

Un elemento è un componente software che rappresenta un'unità di elaborazione del router, ed esegue concettualmente semplici calcoli, come ad esempio decrementare il campo Time-to-live (TTL) di un pacchetto, piuttosto che calcoli complessi, come il routing IP. In generale essi esaminano o modificano i pacchetti in un certo modo. I pacchetti rappresentano le particelle elementari della comunicazione, trasportando le informazioni di rete, che vengono elebore. Durante il funzionamento del dispositivo mappato nella configurazione *Click!* i pacchetti passano da un elemento all'altro attraverso collegamenti chiamati *connessioni*. Ogni elemento è un oggetto C++ che può mantenere una sua autonomia. Dispositivi di processamento, tabelle di instradamento, gestione delle code, conteggi e quant'altro, sono tutte funzioni implementate dagli elementi. Gli elementi hanno cinque importanti proprietà:

1. **Classe dell'elemento**: specifica la struttura dei dati che possono essere trattati dall'elemento e il suo comportamento (quante porte avrà, quali *handlers* supporterà e come elaborerà i pacchetti). In C++ ogni elemento corrisponde ad una sottoclasse della struttura *Element*.

2. **Porte**: Ogni elemento può avere un numero arbitrario di porte di ingresso e uscita. Ogni connessione collega una porta di uscita di un elemento con una porta di entrata di un altro. Il numero di porte di un elemento può essere fisso, oppure può dipendere da una stringa di configurazione, oppure da quante porte sono usate nella particolare configurazione. Infatti ogni porta che viene predisposta deve essere utilizzata da almeno una connessione, altrimenti la configurazione è errata. Le porte possono essere del tipo *Push* (il pacchetto viene fornito dall'elemento), *Pull* (il pacchetto viene richiesto dall'elemento) o *Agnostic* (si adegua al tipo Push o Pull della porta a cui è connessa).

3. **Stringa di configurazione**: La stringa di configurazione è un parametro opzionale, che viene utilizzato per passare agli elementi gli argomenti di configurazione durante la fase di inizializzazione della configurazione, con lo scopo di determinarne lo stato interno e regolarne finemente il comportamento. Dal punto di vista sintattico è costituita da una lista di argomenti separati da virgole. La maggior parte degli argomenti di configurazione appartengono a insiemi limitati di tipi di dati quali, ad esempio numeri interi, o liste di indirizzi IP.

4. **Modalità di interfaccia**: ogni elemento esporta delle modalità di interfaccia a cui gli altri elementi possono accedere. Tutti gli elementi hanno la modalità di interfaccia base, che consente di trasferire i pacchetti; in più alcuni elementi dispongono di ulteriori interfacce. Ad esempio l'elemento *Queue* che implementa una coda FIFO di pacchetti, esporta un'interfaccia che riporta la sua lunghezza corrente.

5. **Handler**: gli *handler* sono modalità di interfaccia esportate a livello di utente piuttosto che agli altri elementi della configurazione router. Ad esempio l'elemento *Queue* menzionato prima ha un *handler* che riporta la sua lunghezza corrente come una stringa ASCII decimale, mentre l'elemento *Counter* mette a disposizione un *handler* che permette all'utente di conoscere il valore corrente del suo contatore.

A.2.2 Connessioni

Ogni connessione rappresenta un percorso possibile per il trasferimento dei pacchetti e collega una porta di uscita di un elemento con una porta di ingresso di un altro, ed ognuna rappresenta un possibile percorso per il trasferimento dei pacchetti tra gli elementi. In un router in esecuzione le connessioni sono rappresentate come puntatori agli oggetti elemento e il passaggio dei pacchetti

lungo una connessione è implementato da una singola chiamata di una funzione virtuale. Graficamente le connessioni vengono rappresentate come frecce che collegano una porta sorgente ad una porta destinazione, indicando la direzione del flusso di pacchetti. Ciascuna connessione collega una porta sorgente ad una porta di destinazione. Una porta sorgente è normalmente una porta d'uscita, mentre una porta di destinazione rappresenta una porta d'ingresso. Nel proseguo spesso si utilizzerà la terminologia di elementi sorgenti ed elementi di destinazione con il significato ovvio. Una configurazione router può essere pensata come un grafo orientato in cui gli elementi ne rappresentano i vertici. Comunque è importante osservare che le connessioni collegano tra loro le porte e non gli elementi stessi, e che ciascun elemento può avere più porte contemporaneamente. Un modello più completo rappresenta le configurazioni router come un grafo orientato in cui le porte rappresentano i vertici. Le porte presentano due tipi di connessioni, quelle ordinarie e quelle interne. Le connessioni interne mostrano come un pacchetto può essere trasmesso da una porta d'ingresso ad una porta d'uscita all'interno di un singolo elemento; il collegamento esistente tra la porta d'ingresso di un elemento e la sua porta d'uscita o, indica che un pacchetto che arriva alla porta d'ingresso i può essere trasmesso sulla porta d'uscita o. Nel modello più semplice ogni elemento presenta una rappresentazione completa dei collegamenti interni, che significa che esistono delle linee interne che collegano ciascun ingresso ad una qualunque uscita. Spesso però questo non sempre è corretto. Infatti, per alcune tipologie di elementi, i pacchetti che arrivano ad una determinata porta d'ingresso possono essere trasmessi solo attraverso un insieme limitato di porte d'uscita, o addirittura attraverso nessuna di esse. Informazioni interne più specifiche, permettono al sistema di decidere quali elementi possono essere raggiunti da una determinata porta; in una lunga esecuzione, esse aiutano ad individuare le proprietà di una configurazione. In generale, se esiste un percorso che collega una porta d'uscita o con una porta d'ingresso i nella rappresentazione grafica della configurazione di un router, diremo che i è successivo ad o (downstream), mentre al contrario, o è precedente ad i (upstream). Questa nozione può essere generalizzata agli elementi.

A.3 Pacchetti Click

Un pacchetto Click consiste in una piccola intestazione di pacchetto e dal reale campo dati del pacchetto IP. L'intestazione del pacchetto punta ai dati. Molte intestazioni possono condividere lo stesso pacchetto dati. Quando si copia un pacchetto, ad esempio tramite l'elemento *Tee*, Click produce una

nuova intestazione che condivide lo stesso pacchetto dati. Gli elementi che modificano i dati devono prima preoccuparsi che non siano condivisi da altre intestazioni; se sono condivisi allora l'elemento deve fare un'unica copia dei dati e cambiare l'intestazione del pacchetto in modo da farla puntare a questa copia.

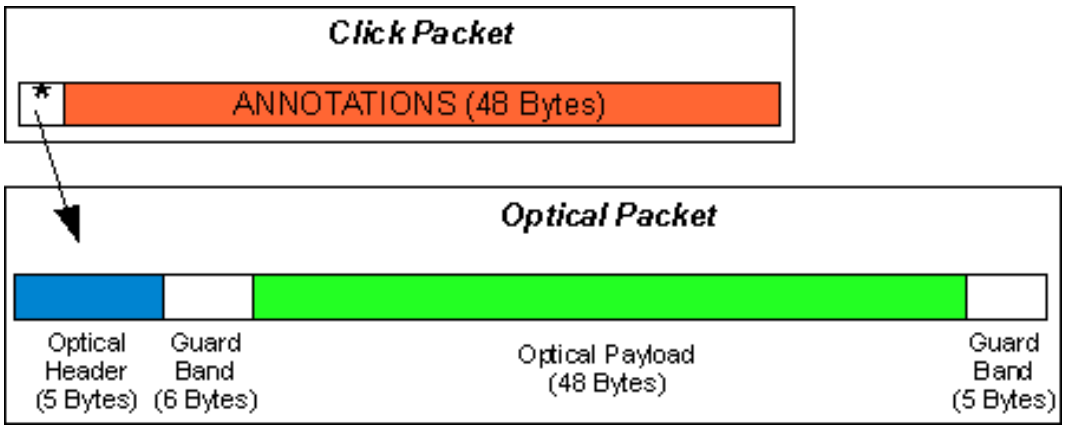

Figura A.2: Una rappresentazione del pacchetto *Click!*

I pacchetti di dati condivisi sono detti perciò copy-on-write. Le intestazioni invece non sono mai condivise, così la loro modifica non provoca mai una copia. Oltre al puntatore ai dati l'intestazione contiene un certo numero di annotazioni, le quali possono essere condivise con Linux, oppure specifiche di *Click!* . Alcune annotazioni contengono informazioni indendenti dai dati (per esempio il tempo in cui il pacchetto è arrivato), mentre altre memorizzano informazioni riguardanti i dati. Per esempio l'elemento *CheckIPHeader* setta l'annotazione IPHeader nei pacchetti IP che transitano. Questa annotazione segnala dove inizia l'intestazione IP e dove inizia il payload, liberando i successivi elementi dall'esaminare il campo 'Length' dell'intestazione IP. Le annotazioni sono memorizzate nell'intestazione del pacchetto in un ordine fisso e statico. I dati del pacchetto sono memorizzati in un singolo buffer di memoria.

A.4 Linguaggio

Il linguaggio del *Click!* descrive testualmente la configurazione del router. Due obiettivi fondamentali guidano la progettazione di un linguaggio, e questi sono:

- leggibilità: un sistema modulare di internetworking risulta facilmente ampliabile, solo se la sua configurazione può essere facilmente letta e modificata

- praticità: La praticità dei tool fondamentalmente significa che dovrebbe essere più facile progettare ed utilizzare degli strumenti specifici per analizzare e modificare i file scritti in *Click!*

Il linguaggio è dichiarativo, ovvero esso semplicemente descrive il grafo relativo ad una configurazione (a differenza dei linguaggi di script, come per esempio i file .tcl di Network Simulator 2 (NS2)). I linguaggi dichiarativi hanno il vantaggio della leggibilità, ma soprattutto possono essere analizzati e modificati in maniera molto più semplice di quanto invece non consentano i linguaggi imperativi.

Il linguaggio è semplice, dato che comprende un numero ridotto di costrutti utilizzabili, preferendo implementare delle estensioni del linguaggio attraverso elementi che avessero degli scopi specifici. Questa scelta limita i meccanismi ed i costrutti che possono essere utilizzati in fase di programmazione, ma consente di ottenere una grande flessibilità.

I programmi scritti in *Click!* e le configurazioni grafiche sono due strutture equivalenti. Ciascuna configurazione corrisponde ad un semplice programma nel linguaggio del *Click!* .

A.4.1 Sintassi

Come abbiamo visto ciascun elemento appartiene ad una classe di elementi, la quale è specificata dal nome ed opzionalmente da una stringa di configurazione. Gli elementi sono connessi attraverso le loro porte d'ingresso e d'uscita. Nel linguaggio specifico del *Click!* , le porte sono distinte attraverso l'uso di numeri, mentre gli elementi sono distinti utilizzando un nome. Ciascun elemento in una configurazione possiede un unico nome, che un utente può specificare in maniera opzionale. Tali elementi individuano e differenziano i vari elementi durante il processo di analisi (anche sintattica eseguita in fase di compilazione), ed permettono inoltre al singolo utente, o ad altri programmi, di accedere ad un particolare elemento, anche in fase di esecuzione (lancio della configurazione).

Il comando utilizzato per effettuare la connessione crea una connessione dalla porta d'uscita port1 dell'elemento 'name1' con la porta d'ingresso 'port2' dell'elemento 'name2'. Gli elementi devono essere dichiarati prima di essere utilizzati nelle connessioni. Ciascuna configurazione può essere descritta attraverso solo questi due costrutti sintattici; ma delle strutture sintattiche aggiuntive possono essere utilizzate per rendere visivamente più semplici le diverse configurazioni.

Appendice B

Appendice B: Installazione in laboratorio

L'esperienza è il tipo di insegnante più difficile. Prima ti fa l'esame poi ti spiega la lezione.

Anonimo.

In questa seconda appendice presenteremo il lavoro di installazione e configurazione delle macchine del laboratorio del DEIS al fine di documentare i passi svolti per testare l'implemntazione del modello e ottenere i risultati sperimentali.

B.1 Installazione delle macchine

Si è scelto di approntare tre macchine per l'esecuzione dei test, in vista anche di una futura configurazione che incrementi le prestazioni dell'emulazione. A tale scopo sono state installate:

- 2 Macchine desktop Dell® Optiplex GX270
- 1 Macchina server Dell® PowerEdge Server 1600SC

Tutte e tre i calcolatori sono inoltre dotati di due interfacce di rete Intel® con supporto per lo standard Gigabit Ethernet, in modo da non costituire colli di bottiglia nella modellazione del nodo.

B.2 Installazione del sistema operativo

Come sistema operativo si è optato per l'installazione della distribuzione Fedora core 12, per testare il software *Click!* su una piattaforma differente da quella di sviluppo e verificare ulteriormente la compatibilità dell'implementazione realizzata.

Figura B.1: Il logo della distribuzione Fedora

Successivamente alla fase di installazione eseguita mediante l'assistente personalizzato della distribuzione (in cui si è avuto cura di selezionare i tool e le librerie per lo sviluppo del software, tra cui i compilatori g++)

B.2.1 Ottimizzazione

Al fine di ottimizzare le risorse a disposizione del sistema operativo, e quindi in definitiva a disposizione dell'istanza *Click!* che vi verrà eseguita, è stata definita una configurazione di partenza del sistema operativo con un numero di servizi e componenti ridotti al minimo. Si è dunque provveduto a modificare il file grub.conf (il file di configurazione del *bootloader*) in modo da inizializzare il sistema a *runlevel* 3, una modalità in cui è disabilitata oltra l'interfaccia grafica, la gran parte dei servizi accessori. Inoltre si è provveduto a disabilitare ulteriori servizi non indispensabili al nostro scopo come il servizio di stampa (cups), il bluetooth (bluetooth daemon), i servizi legati alle remote procedure call (rpc) e diversi altri. Alla fine di questo passo il sistema sarà leggero e in grado di dedicare tutte le risorse al software *Click!* in esecuzione sulla macchina.

B.2.2 Installazione delle dipendenze

Come ogni software creato per linux anche il *Click!* si appoggia a librerie o eseguibili esterni che però non sempre sono compresi nella distribuzione di base. Per ovviare a questo problema vengono riportati in tabella i pacchetti necessari e/o complementari all'installazione del software di modellazione:

Pacchetti e dipendenze *Click!* :	
Compiler:	gcc
Compiler extension:	gcc ada (gnat)
Perl extension:	pcap
Compiler:	gcc-c++ (g++)
Compiler:	gcc-info
Compiler:	gcc-objc (gobjc++)
Perl:	perl5
Awk:	gawk
Documentation:	texinfo
Dvi:	texi2dvi
Make:	automake
Config:	autoconf
Gtk:	libgtk2-dev
Graphics:	graphviz

Tabella B.1: Pacchetti e dipendenze *Click!*

B.3 Installazione del software *Click!*

Di seguito vengono presentati i principali passi per l'installazione del software.

- **Passo 1**: Per ottenere il codice sorgente del *Click!* ci si è affidati al *repository* del progetto in cui è possibile trovare sempre l'ultima versione stabile e aggiornata del software. Inoltre in questo modo è possibile ottenere una versione *Click!* per l'esecuzione a livello *kernel* in modalità *patchless* ovvero senza dover ricompilare il nucleo del sistema operativo per l'inserimento della configurazione *Click!* come modulo del *kernel*. Per ottenere i sorgenti dunque è necessario innanzitutto installare il Concurrent Versions System (CVS) denominato *git*:

```
yum install git
```

- **Passo 2**: A questo punto, installato il CVS occorre reperire i sorgenti del *Click!* mediante il comando *clone* per inserirli in una *directory* a nostra scelta specificata dal parametro DIR:

```
git clone git://read.cs.ucla.edu/git/click DIR
```

- **Passo 3**: A questo punto vengono scaricati i sorgenti *Click!* , compresa la *change history* in circa 37 MB di spazio su disco. A questo punto, spostandoci nella cartella dei sorgenti eseguiamo lo script di configurazione con le opzioni per includere la compilazione dell'eseguibile kernel-level patchless (–enable-fixincludes) e per la compilazione dei file C++ creati

per la configurazione del router ottico precedentemente posizionati nella cartella DIR/elements/local:

```
./configure --enable-fixincludes --enable-local
```

- **Passo 4**: A questo punto non rimane che compilare il *Click!* con il comando:

  ```
  make
  ```

- **Passo 5**: Ed infine installare gli eseguibili con il comando:

  ```
  make install
  ```

- **Passo 6**: A questo punto il software *Click!* è correttamente installato nel sistema ed è possibile eseguire una qualsiasi configurazione *Click!* . Ad esempio è possibile eseguire una configurazione di test con il comando:

  ```
  click conf/test.click
  ```

B.3.1 Installazione di Clicky GUI

Il *Click!* è dotato di una interfaccia grafica denominata Clicky GUI che è compresa nel pacchetto sorgente ottenuto in precedenza.

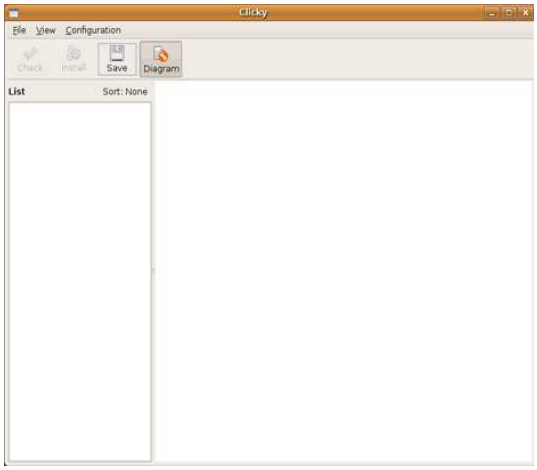

Figura B.2: L'interfaccia grafica Clicky

Per compilarla ed eseguirla è necessario eseguire alcuni semplici passi.

- **Passo 1**: Innanzitutti configuriamo clicky eseguendo dalla cartella DIR/apps/clicky il comando:

  ```
  autoreconf -i
  ```

- **Passo 2**: A questo punto, lanciamo lo script di configurazione:

  ```
  ./configure
  ```

- **Passo 3**: A questo punto non rimane che compilare il *Click!* con il comando:

  ```
  make install
  ```

- **Passo 4**: A questo punto non rimane che lanciare l'applicazione:

  ```
  clicky
  ```

Ringraziamenti

Lo studio: strumento per costruire la propria libertà, educazione dell'ingegno e della creatività al lavoro, ma soprattutto occasione privilegiata di capire la vita.

Enrico Palandri, scrittore italiano.
(1956 - vivente)

Se ti capiterà di leggere queste pagine, allora vorrà dire che ho avuto l'onore di conoscerti. Così voglio dirti che porterò sempre con me i bei momenti passati insieme, le belle serate trascorse e le risate che ci siamo fatti, spesso, anche a lezione. E a prescindere dalle occasionali incomprensioni, dai litigi, o dalla distanza che può averci separato per qualche periodo o in alcune idee o pensieri, grazie di cuore, perchè in tutto questo lavoro, in tutte queste pagine, c'è anche un pò di te.

I primi sinceri e profondi ringraziamenti voglio dedicarli senza retorica alla mia famiglia, alla mia Mamma, al mio Papà e a mio fratello Omar senza i quali, devo riconoscerlo, non avrei raggiunto questo prestigioso traguardo. Sempre presenti e vicini, sempre le parole giuste. Avete creduto in me più di quanto lo facessi io. Troppo spesso con i miei comportamenti e le mie lamentele vi ho visto chiedervi se eravate dei cattivi genitori, se non stavate sbagliando con me. La risposta è sempre stata no, sempre, e siate orgogliosi perchè questo risultato è anche vostro.

A mio fratello Omar dedico due righe in più, per sottolineare il piacere e l'onore che ho avuto nel condividere con lui questa lunga esperienza. Il bene che ti vuole un fratello forse non si può misurare, ma io ho avuto l'occasione di provarlo, giorno dopo giorno, in questi lunghi otto anni insieme. Grazie tato.

Ai nonni infine il mio pensiero va spontaneo; spero di avervi reso fiero di me tanto quanto mi mancano, ogni giorno di più, i vostri abbracci e i vostri sorrisi. Non dimenticherò mai i vostri insegnamenti, che solo con il passare degli anni, purtroppo, riscopro sempre più importanti e paterni. Quello che mi rattrista è non potervi rendere merito di tutto ciò. Grazie Nonno Peppe, Nonno Angelo, Nonna Anita e Nonna Fenisia.

Chi merita poi un ringraziamento speciale, fuori da ogni qualsivoglia classifica, perchè sempre prima è e sarà nel mio cuore, è la mia fidanzata, Susanna. Sei sempre stata al mio fianco, nei momenti belli e ancor di più in quelli brutti, mi hai spronato e confortato ogni giorno. Nessun aggettivo descriverebbe tutto quello che sei per me e queste righe non sono sufficenti per spiegare quanto tu sia speciale. Spero però basti una vita insieme. Grazie, amore mio.

Un saluto sincero va anche a quelle persone che solo negli ultimi anni sono entrati a far parte della mia famiglia, ma non per questo sono meno importanti. A Giancarlo, Rosina e Giorgio, e alla mia cognatina Catia voglio dire grazie per la naturalezza e l'affetto con il quale mi avete sempre accolto nelle vostre case e mi avete fatto sentire parte delle vostre famiglie.

I ringraziamenti non sarebbero completi se non citassi due pilastri fondamentali del mio cuore, i miei due patatini, Porthos e Ares. Spesso siamo lontani, a volte non vi coccolo abbastanza o fa troppo freddo per uscire ma vorrei tanto sapeste, che come mi fate tornare il sorriso voi, non ci riesce nessuno. Anche se non potrete mai leggere queste parole il vostro posto è qui tra i miei affetti più cari.

Al mio relatore, la professoressa Carla Raffaelli, e ai miei correlatori, i dottori Michele Savi e Walter Cerroni vanno i miei sinceri ringraziamenti per la serietà, la profesionalità e l'aiuto mostratomi in questo progetto, attraverso le sue difficoltà, i suoi alti e bassi, sempre fiduciosi delle mie capacità e disponibili al dialogo e all'assistenza. Spero di aver ricambiato la vostra fiducia con il lavoro svolto.

Ai miei amici più cari, cui mi lega un'amicizia lunga una vita, Stefano e Alessandro. Con loro ho condiviso tutta la mia adolescenza, le amicizie, i momenti belli e quelli brutti, i miei ed i loro, quando abitavamo vicini, e quando siamo stati lontani, insomma sempre. Ne abbiamo passate tante insieme, tutte indimenticabili. Grazie di cuore, ragazzi.

Ai miei coinquilini, passati e presenti, con cui è stato memorabile vivere tutti insieme come una grande famiglia, a partire dal mio amico Claudio, il piccolo Mattia, Salvo, Paolo, il nonno Patric e il mio *Maestro-Zio* Andrea. Di questi anni ricorderò sopratutto le serate insieme, gli scherzi e le risate che non sono mai mancate. La nostalgia più grande di questa esperienza è e sarà sempre quella di non avervi più in giro per casa. Mi mancherete.

A Eleonora, Simone, Valentina e Marco, un ringraziamento particolare per avermi accolto nelle loro amicizie, nelle loro serate e infine, complice cupido, nelle loro case. Gran parte di questi ultimi tre anni l'ho passata con voi, condividendo le ansie, le paure ma anche qualche soddisfazione (e qualche pettegolezzo!). Anche a queste belle serate saranno legati i miei ricordi universitari. Grazie.

Ad un gruppo speciale, ormai con un pò di trasferte alle spalle e nuove amicizie nate condividendo una passione ed un orgoglio tutto piceno. Agli amici *Esiliati* mi legano i ricordi più intensi degli ultimi anni, i chilometri fatti insieme, le delusioni ma anche clamorose soddisfazioni, sempre indelebili, di un esperienza unica. Il mio vanto è essere uno di voi. Al mio amico Luca de Filippis, e al nucleo storico formato da Marco Minelli, Tiziano Caponi, ai numerosi altri che si sono aggiunti nel tempo: Gianmarco Rendina, Davide Ferretti, Alessandro Ricci, Stefano Virgili, Francesco Aurini, Enrico Seghetti, Matteo Sabatini, Antonio Angelelli, Piergiulio Manardi, Iacopo Mattia Perozzi, Roberto Battilana. Siete pronti per una nuova trasferta, ragazzi?

Ai miei fedeli compagni di gradoni, nelle partite casalinghe del magico Picchio, e ai miei amici ascolani, che aspettano da tempo questa notizia. Voglio così ringraziare Carlo e Stefano Diamanti, Roberta Bordoni, Sara Abeti, Stefano Ciannavei e Iole D'Angelo e tanti altri ...

Agli amici conosciuti qui e con cui ho diviso quest'esperienza accademica, Christian Florio e Davide Gasbarro, Renato Grottesi, Costanzo Di Maria, Francesco Achille, Leonardo D'Apote, Marco Damiani, Mauro Lopopolo, Francesco Fazzini, Fabio Ciotoli, Andrea Cirri, Fausto Fusaro, Francesco Ceravolo e Chiara Gualtieri. Indimenticabili i momenti a lezione, i pomeriggi di studio, le tensioni degli esami fatti e le gioie di quelli passati, insieme.

Alla mia città, Ascoli Piceno, la cui nostalgia fa sempre capolino nel mio cuore e all'Ascoli Calcio, passione vera e unico svago nei miei momenti difficili, orgoglio e soddisfAzione.

A tutti Voi, grazie di cuore.
Raul.

Bibliografia

[1] Frank Mittelbach and Michel Goossens "*The LaTeX Companion (Second Edition) : Tools and Techniques for Computer Typesetting*" Addison Wesley, 2004.

[2] AA. VV. "*Wikipedia: The Free Encyclopedia*" Wikimedia Foundation http://www.wikipedia.org/

[3] AA. VV. "*Linux FEDORA. Guida professionale*" Apogeo, 2005. http://www.apogeonline.com/libri/88-503-2314-X/scheda?id=8KnyX6mC

[4] C.Raffaelli, M. Savi, A. Stavdas "*Multistage Shared-Per-Wavelength Optical Packet Switch: Heuristic Scheduling Algorithm and Performance*" IEEE/OSA Journal of Lightwave Technology, Vol. 27, Issue 5, pages 538-551, March 2009.

[5] C.Raffaelli, M. Savi, W. Cerroni "*Modular Design of Programmable Multi-Granular Optical Switching Node*"

[6] Herbert Schildt "*C++ La guida completa*" McGraw-Hill, 1995.

[7] Biswanath Mukherjee "*Optical WDM Networks*" Springer, 2005.

[8] Tarek S. El-Bawab "*Optical Switching*" Springer, 2005.

[9] Rajiv Ramaswami, Kumar N. Sivarajan "*Optical Networks, a practical perspective (Second Edition)*" Morgan-Kaufmann, 2001.

[10] G. S. Zervas, M. De Leenheer, L. Sadeghioon, D. Klonidis, Y. Qin, R. Nejabati, D. Simeonidou, C. Develder, B. Dhoert, and P. Demeester, ÒMulti-Granular Optical Cross-Connect: Design, Analysis and DemonstrationÓ, *Journal on Selected Areas in Communications (JSAC), IEEE*, vol. 27, no. 4, April 2009.

[11] F. Callegati, A. Campi, G. Corazza, D. Simeonidou, G. Zervas, Y. Qin, R. Nejabati, SIP-empowered Optical Networks for Future IT Services and Applications, *IEEE Communications Magazine*, Vol. 47, Issue 5, pp. 48-54, May 2009.

[12] L. Wosinska, D. Simeonidou, A. Tzanakaki, C. Raffaelli C. Politi, Optical Networks for the Future Internet: Introduction, *IEEE/OSA Journal of Optical Communications and Networking*, Vol. 1, Issue 2, pp. FI1ÐFI3, July 2009.

[13] G. Zervas, R. Nejabati, D. Simeonidou, C. Raffaelli, M. Savi, C. Develder, M. De Leenheer, Dider Colle, M. Schiano, Programmable Multi-Granular Optical Networks: Requirements and Architecture, *in proceedings of Broadnets 2009*, Madrid, Spain, 14-16 September 2009.

[14] G. Zervas et al., Demonstration of Novel Multi-Granular Switch Architecture on an Application-Aware End-to-End Multi-Bit Rate OBS Network Testbed, *European Conference on Optical Communication (ECOC 2007)*, PostDeadline, PDS 3.2, Berlin, Germany, Sept. 2007.

[15] B. Martini, V. Martini, F. Baroncelli, K. Torkman, P. Castoldi, - Application-Driven Control of Network Resources in Multiservice Optical Networks, *Journal of Optical Communications and Networking*, Vol. 1, Issue 2, pp. A270ÐA283, July 2009.

[16] Y. Qin et Al, Service-Oriented Multi-Granular Optical Network Testbed, *in proceedings of IEEE/OSA OFC 2009*x, Optical Fiber Communication Conference, San Diego, USA, 22-26 March, 2009.

[17] E. Kohler, R. Morris, B. Chen, J. Jannotti, M.F. Kaashoek, The Click Modular Router, *ACM Transactions on Computer Systmes*, 18(3), 2000, pp. 263-297.

[18] Yu Ben, Qian Ying Tang "*Optical Packet Switching*", May 2, 2006.

[19] AA. VV. "*An Optical Packet Switch Based on WDM Technologies*"

[20] AA.VV. "*Optical Packet Switching*" Cambridge Press.

[21] AA. VV. "*Gateway for India*"
http://www.gatewayforindia.com/technology/opticalfiber.htm

[22] K.Kitayama, M.Koga, H.Morikawa, S.Hara, and M.Kawai "*Optical Burst Switching Network Test bed in Japan*", in proceedings of Optical Fiber

Communication Conference (OFC2005), Anaheim, USA, Mar. 2005, PaperOFA6.

[23] O.Liboiron-Ladouceur, A.Shacham, B.A.Small, B.G.Lee, H.Wang, C.P.Lai, A.Biberman, K.Bergman "*The Data Vortex Optical Packet Switched Interconnection Network*", Journal of Lightwave Technology, Vol.26, Issue13, pp.1777-1789, July 2008.

[24] Carla Raffaelli, Giovanni Schembra "*Introduzione all'uso dell'ambiente Click! per la realizzazione di router IP*", Rapporto tecnico DEIS - Università di Bologna DEIS-NET-03-001

[25] G.Calarco, C.Raffaelli, "*Implementation of Implicit QoS Control in a modular software router context*", QoS-IP2005, Catania, Italy, February 2005.

[26] J.Somers, P.Barford, M.Crovella "*Router Primitives for Programmable Active Measurement*", Proceedings of ACM SIGCOMM PRESTO Workshop, August 2009.

[27] L.Zanolin, C.Mascolo, W.Emmerich "*Model checking programmable router configurations*", UCL Research Note (02/23).

Elenco degli acronimi

ASCII American Standard Code for Information Interchange. Ovvero Codice Standard Americano per lo Scambio di Informazioni è un sistema di codifica dei caratteri a 7 bit comunemente utilizzato nei calcolatori, proposto dall'ingegnere dell'IBM Bob Bemer nel 1961, e successivamente accettato come standard dall'ISO (ISO 646).

ATM Asynchronous Transfer Mode. E' un protocollo di rete a commutazione di cella che incapsula il traffico in celle a lunghezza fissa (53 byte) invece che in pacchetti a lunghezza variabile come nelle reti a commutazione di pacchetto (ad esempio IP).

BaS Broadcast-and-Select. Tipologia di architettura per router ottici che prevede la duplicazione dei percorsi provenienti dagli ingressi verso tutte le uscite possibili con una successiva selezione dei percorsi effettivamente da abilitare.

CE Control Element. Componente dedicato alla gestione dei segnali di controllo del router ottico.

CVS Concurrent Versions System E' un sistema di controllo versione, mantiene cioè al corrente di tutto il lavoro e di tutti i cambiamenti in un insieme di file, tipicamente è l'implementazione di un software in via di sviluppo, in progetto, e permette a molti sviluppatori (potenzialmente distanti) di collaborare. CVS è divenuto popolare nel mondo del software libero ed è rilasciato sotto la GNU General Public License.

CWDM Coarse Wavelength Division Multiplexing. Variante della multiplazione WDM utilizzata nei sistemi di comunicazione ottica. Nel coarse WDM la separazione tra le lunghezze d'onda usate è maggiore che nel convenzionale e nel DWDM, in modo da poter utilizzare componenti ottici meno sofisticati e quindi meno costosi.

DEIS Dipartimento di Elettronica, Informatica e Sistemistica..

DWDM Dense Wavelength Division Multiplexing. Variante della multiplazione WDM utilizzata nei sistemi di comunicazione ottica. Il Dense WDM usa la stessa finestra di trasmissione ma con minore separazione tra i canali, arrivando a 31 canali a intervalli di 50 GHz.

EDFA Erbium-Doped Fiber Amplifiers. Sono amplificatori ottici che usano un fibra ottica drogata come mezzo attivo per amplificare un segnale ottico. Il segnale che si vuole amplificare ed un segnale di pompa vengono multiplati in una fibra drogata ed il segnale risulta amplificato mediante l'interazione con gli ioni del drogante.

EO Electrical-Optical. Conversione di un segnale dal dominio elettromagnetico (livelli di tensione variabili, interpretabili come valore zero o uno) a quello ottico (assenza o presenza di luce).

EPS Electronic Packet Switching. Con questo termine si intende il processo di elaborazione e commutazione dei pacchetti all'interno dei router elettronici.

FDL Fiber Delay Line. Componente in grado di ritardare nel tempo il segnale ottico in ingresso di un valore T fissato.

FDM Frequency Division Multiplexing. Ovvero multiplazione a divisione di frequenza, è una tecnica di condivisione di un canale di comunicazione secondo la quale un canale trasmissivo è diviso in sottocanali, ognuno costituito da una banda di frequenza separata. Questo rende possibile la condivisione dello stesso canale da parte di diversi dispositivi che possono comunicare contemporaneamente.

FE Forwarding Element. Componente dedicato alla gestione dell'inoltro dei pacchetti nel router ottico.

FIFO First-In First-Out. Politica di gestione delle code, in cui il primo elemento ad entrare nella coda sarà il primo ad uscirne.

FM Forwarding Module. Componente dedicato all'attuazione dell'inoltro dei pacchetti nel router ottico.

ForCES Forwarding and Control Element Separation. Raccomandazione dell'IEEE per la separazione del piano di controllo ed inoltro nei router di nuova generazione.

FT Forwarding Table. Tabella contenente le informazioni di inoltro per una determinata label MPLS.

FTWC Fixed-input Tunable-output Wavelength Converter. Convertitore che accetta solo un determinata lunghezza d'onda fissa in ingresso mentre permette di convertirla verso l'uscita su una qualsiasi altra lunghezza d'onda.

GMPLS Generalized Multi-Protocol Label Switching. MPLS è nato principalmente per garantire alte performance di inoltro del traffico, sia IP che di livello 2, ed è stato oggetto di estensioni per garantire la creazione di percorsi anche su reti non nativamente IP, quali reti SDH e WDM. In questa forma è noto come Generalized MPLS o G-MPLS. Il concetto di label è stato ampliato includendo anche identificativi di diverso tipo, quali l'associazione a numero di timeslot in trama SDH oppure frequenze di wavelenght per i sistemi WDM.

GNU GNU is Not Unix. Il progetto GNU lanciato nel 1983 da Richard Stallman, si basa su una gestione particolare dei diritti d'autore sul software, secondo la definizione di software libero (contrapposta a software proprietario).

IEEE Institute of Electrical and Electronic Engineers Istituto degli ingegneri elettrici ed elettronici, nacque il 1^{o} gennaio 1963 con lo scopo principale di cercare nuove applicazioni e teorie nella scienza elettrotecnica, elettronica, informatica, meccanica e biomedica; a questo scopo organizza conferenze e dibattiti tecnici in tutto il mondo, pubblica testi tecnici e sostiene programmi educativi. Si occupa inoltre di definire e pubblicare standard in tali campi.

IETF Internet Engineering Task Force. E' una comunità aperta di tecnici, specialisti e ricercatori interessati all'evoluzione tecnica e tecnologica di Internet. Ciò che differenzia IETF dagli Enti di standardizzazione più tradizionali è la sua struttura aperta: il lavoro viene svolto da gruppi di lavoro (working groups) che operano soprattutto tramite Mailing list, aperte alla partecipazione di chiunque sia interessato, e che si riuniscono tre volte l'anno. I gruppi di lavoro si occupano ciascuno di uno specifico argomento e sono organizzati in aree (protocolli applicativi, sicurezza, ...).

IF Input Fiber. Fibra di ingresso.

IP Internet Protocol. E' un protocollo di interconnessione di reti (Inter-Networking Protocol), nato per interconnettere reti eterogenee per tecnologia, prestazioni, gestione. E' di tipo connection-less ed è classificato ISO/OSI livello 3 (rete).

ISO International Organization for Standardization E' la più importante organizzazione a livello mondiale per la definizione di norme tecniche. Fondata il 23 febbraio 1947, ha il suo quartier generale a Ginevra in Svizzera.

ISP Internet Service Provider. Un fornitore di servizi internet è una struttura commerciale o un'organizzazione che offre agli utenti (residenziali o imprese) servizi inerenti Internet i principali dei quali sono l'accesso alla rete stessa e la posta elettronica.

KEOPS KEys to Optical Packet Switching. Progetto europeo per lo studio delle soluzioni in ambito del trasferimento ottico delle informazioni.

LED Light Emitting Diode. Diodo ad emissione luminosa, sfrutta le proprietà ottiche di alcuni materiali semiconduttori per produrre fotoni a partire dalla ricombinazione di coppie elettrone-lacuna.

LL Label Lookup. Dispositivo di interrogazione e risoluzione degli indirizzi nelle label usate in ambienti tipo MPLS.

LS Label Swap. Dispositivo di sostituzione degli indirizzi nelle label usate in ambienti tipo MPLS.

MAN Metropolitan Area Network. La rete in area metropolitana è una tipologia di rete di telecomunicazioni con un'estensione limitata a perimetro metropolitano. Storicamente le MAN sono nate per fornire servizi di tv via cavo alle città dove c'era una cattiva ricezione terrestre. In pratica un'antenna posta su una posizione favorevole, distribuiva poi il segnale alle case mediante cavo. Quando il fenomeno Internet è esploso, queste società hanno ben pensato di diffondere la comunicazione internet anche attraverso il cavo TV utilizzando la stuttura preesistente. Tipicamente questa struttura, attualmente, utilizza la fibra ottica come mezzo di collegamento.

MEMS Micro-Electro-Mechanical Systems. Sigla che indica quello che la tecnologia del microscopico ha prodotto (si intende qui che la dimensione media degli oggetti considerati sia di un micrometro). I microsistemi elettromeccanici non sono altro che un insieme di dispositivi di varia natura (meccanici, elettrici ed elettronici) integrati in forma altamente miniaturizzata su uno stesso substrato di silicio, che coniugano le proprietà elettriche degli integrati a semiconduttore con proprietà opto-meccaniche.

MIT Massachusetts Institute of Technology. E' una delle più importanti università di ricerca del mondo, con sede a Cambridge, nel Massachusetts.

MPLS Multi Protocol Label Switching. E' una tecnologia per reti IP che permette di instradare flussi di traffico multiprotocollo tra origine (Ingress Node) e destinazione (Egress Node) tramite l'utilizzo di identificativi (label) tra coppie di router adiacenti ed operazioni semplici sulle etichette stesse.

MST Micro Systems Technology. Sinonimo dei Micro-Electro-Mechanical Systems (MEMS).

NS2 Network Simulator 2. Software per la simulazione di architetture di rete.

OBS Optical Burst Switching. Paradigma di commutazione a *burst*, prevede il passaggio prioritario di flussi di dati contrassegnati da un qualche tipo di livello di servizio o indice di priorità.

OCS Optical Circuit Switching. Paradigma di commutazione a circuito, prevede l'instaurazione di un canale di comunicazione dedicato ed esclusivo tra il mittente e il ricevente.

OE Optical-Electrical. Conversione di un segnale dal dominio ottico (assenza o presenza di luce) a quello elettromagnetico (livelli di tensione variabili, interpretabili come valore zero o uno).

OEO Optical-Electrical-Optical. Serie di conversioni di un segnale dal dominio ottico (assenza o presenza di luce) a quello elettromagnetico (livelli di tensione variabili, interpretabili come valore zero o uno) per il processamento mediante dispositivi elettronici. Successivamente il segnale viene di nuov convertito dal dominio elettromagnetico a quello della luce per tornare a viaggiare all'interno della rete ottica.

OF Output Fiber. Fibra di uscita.

OLS Optical Label Switching. Tecnica di commutazione basata sulla interpretazione di etichette apposte in incipit al traffico. In base al valore espresso in queste etichette il traffico viene rediretto o meno verso una determinata interfaccia di uscita.

OLPS Optical Label Packet Switching. Termine equivalente per indicare la tecnica OLS.

OPATM Optically Transparent Asynchronous Transfer Mode. Proposta di protocollo trasparente alla rete ATM su fibra ottica.

OPNET OPerations NETwork. Software per la simulazione di architetture di rete.

OPS Optical Packet Switching. Paradigma di commutazione a pacchetto in cui le informazioni di più canali di comunicazioni viaggiano mediante lo stesso collegamento, confinate in pacchetti che usano il mezzo trasmissivo per frazioni di tempo distinte, realizzando di fatto la condivisione del canale.

OSI Open Systems Interconnection E' uno standard stabilito nel 1978 dall'ISO, il principale ente di standardizzazione internazionale, che stabilisce una pila di protocolli in 7 livelli di un modello standard di riferimento per l'interconnessione di sistemi di computer.

PLP Packet Loss Probability. Probabilità di perdita di pacchetti in un nodo della rete. E' il risultato della divisione tra il valore dei pacchetti in ingresso al nodo meno il valore dei pacchetti in uscita (quindi il valore dei pacchetti persi nelle operazioni di scheduling) e il valore dei pacchetti in ingresso al nodo.

QoS Quality of Service. Termine per indicare i parametri usati per caratterizzare la qualità del servizio offerto dalla rete (ad esempio perdita di pacchetti, ritardo), o gli strumenti per ottenere una qualità di servizio desiderata. La qualità del servizio è normalmente correlata negativamente con il traffico offerto alla rete, e positivamente con le risorse impegnate per realizzare e gestire la rete.

RAM Random Access Memory. La memoria ad accesso casuale, è una tipologia di memoria informatica caratterizzata dal permettere l'accesso diretto a qualunque indirizzo di memoria con lo stesso tempo di accesso. La memoria ad accesso casuale si contrappone alla memoria ad accesso sequenziale e alla memoria ad accesso diretto rispetto alle quali presenta tempi di accesso sensibilmente inferiori motivo per cui è utilizzata come memoria primaria.

RFC Request For Comments. E' un documento che riporta informazioni o specifiche riguardanti nuove ricerche, innovazioni e metodologie dell'ambito informatico o, più nello specifico, di Internet. Attraverso l'Internet Society gli ingegneri o gli esperti informatici possono pubblicare dei memorandum, sottoforma di RFC, per esporre nuove idee o semplicemente delle informazioni che una volta vagliati dall'IETF possono diventare degli standard Internet.

SDM Space Division Multiplexing. La multiplazione a divisione di spazio, è una tecnica di condivisione di un canale di comunicazione secondo la quale ogni dispositivo abbia un canale separato di comunicazione e uno spazio di guardia tra gli altri mittenti.

SM Switching Matrix. Matrice di commutazione. E' il componente che realizza il passaggio delle informazioni dalle interfacce di input a quelle di output mediante l'attuazione dei suoi dispositivi interni, che caratterizzano l'architettura, secondo le direttive imposte dal piano di controllo.

SOA Semiconductor Optical Amplifier. Dispositivi di amplificazione del segnale ottico i quali utilizzano un semiconduttore per fornire il guadagno.

SPL Shared-Per-Link. Architettura di tipo shared in cui i WC vengono condivisi per interfaccia di ingresso.

SPN Shared-Per-Node. Architettura di tipo shared in cui i WC vengono condivisi per nodo.

SPW Shared-Per-Wavelength. Architettura di tipo shared in cui i WC vengono condivisi per lunghezza d'onda di ingresso.

TCP Transmission Control Protocol. E' un protocollo di livello di trasporto della suite di protocolli Internet. Può essere classificato al livello trasporto (OSI level 4) del modello di riferimento OSI, e di solito è usato in combinazione con il protocollo di livello rete (OSI level 3) IP. Il TCP è stato progettato per utilizzare i servizi del protocollo IP, che non offre alcuna garanzia in ordine alla consegna dei pacchetti, al ritardo, alla congestione, e costruire un canale di comunicazione affidabile tra due processi applicativi. Il canale di comunicazione è costituito da un flusso bidirezionale di byte.

TDM Time Division Multiplexing. La multiplazione a divisione di tempo, è una tecnica di condivisione di un canale di comunicazione secondo la quale ogni dispositivo ottiene a turno l'uso esclusivo dello stesso per un breve lasso di tempo (tipicamente 125 micro secondi).

TOS Type Of Service. Bit relativi al tipo di servizio desiderato che si trovano nell'intestazione IPv4 per distinguere diversi tipi di datagrammi.

TTL Time-to-live. E' un campo dell'header del protocollo IP e di MPLS, che determina il numero massimo di router che possono essere attraversati da un pacchetto.

TWC Tunable-output Wavelength Converter. Convertitore in grado di convertire la lunghezza d'onda in ingresso su una qualsiasi altra lunghezza d'onda in uscita. Esistono diveri tipi di TWC come ad esempio i Fixed-input Tunable-output Wavelength Converter (FTWC) o i Tunable-input Tunable-output Wavelength Converter (TTWC).

TTWC Tunable-input Tunable-output Wavelength Converter. Convertitore che accetta in ingresso una qualsiasi lunghezza d'onda e permette di convertirla verso l'uscita su una qualsiasi altra lunghezza d'onda.

UDP User Datagram Protocol. E' uno dei principali protocolli della suite di protocolli Internet a livello di trasporto a pacchetto, usato di solito in combinazione con il protocollo IP. l'UDP è un protocollo di tipo connectionless, inoltre non gestisce il riordinamento dei pacchetti né la ritrasmissione di quelli persi, ed è perciò generalmente considerato di minore affidabilità. È in compenso molto rapido ed efficiente per le applicazioni leggere o time-sensitive. Ad esempio, è usato spesso per la trasmissione di informazioni audio o video. Dato che le applicazioni in tempo reale spesso richiedono un ritmo minimo di spedizione, non vogliono ritardare eccessivamente la trasmissione dei pacchetti e possono tollerare qualche perdita di dati, il modello di servizio TCP può non essere particolarmente adatto alle loro caratteristiche.

VLAN Virtualized Local Area Network. Indica un insieme di tecnologie che permettono di segmentare il dominio di broadcast, che si crea in una rete locale (tipicamente IEEE 802.3) basata su switch, in più reti non comunicanti tra loro. Le applicazioni di questa tecnologia sono tipicamente legate ad esigenze di separare il traffico di gruppi di lavoro o dipartimenti di una azienda, per applicare diverse politiche di sicurezza informatica.

WC Wavelength Converter. Dispositivo in grado di convertire la lunghezza d'onda di un segnale ottico modulato in WDM.

WDM Wavelength Division Multiplexing. Un tipo di multiplazione utilizzato nei sistemi di comunicazione ottica. Per modulare diversi canali su una stessa fibra ottica si usano diverse portanti di differenti lunghezze d'onda, una per ogni canale, e per la singola portante si usa la modulazione di intensità. In questo modo è possibile sfruttare la grande banda ottica disponibile.