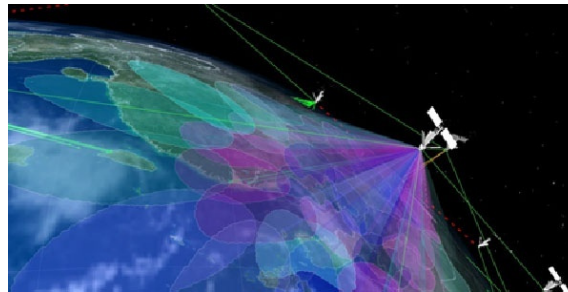
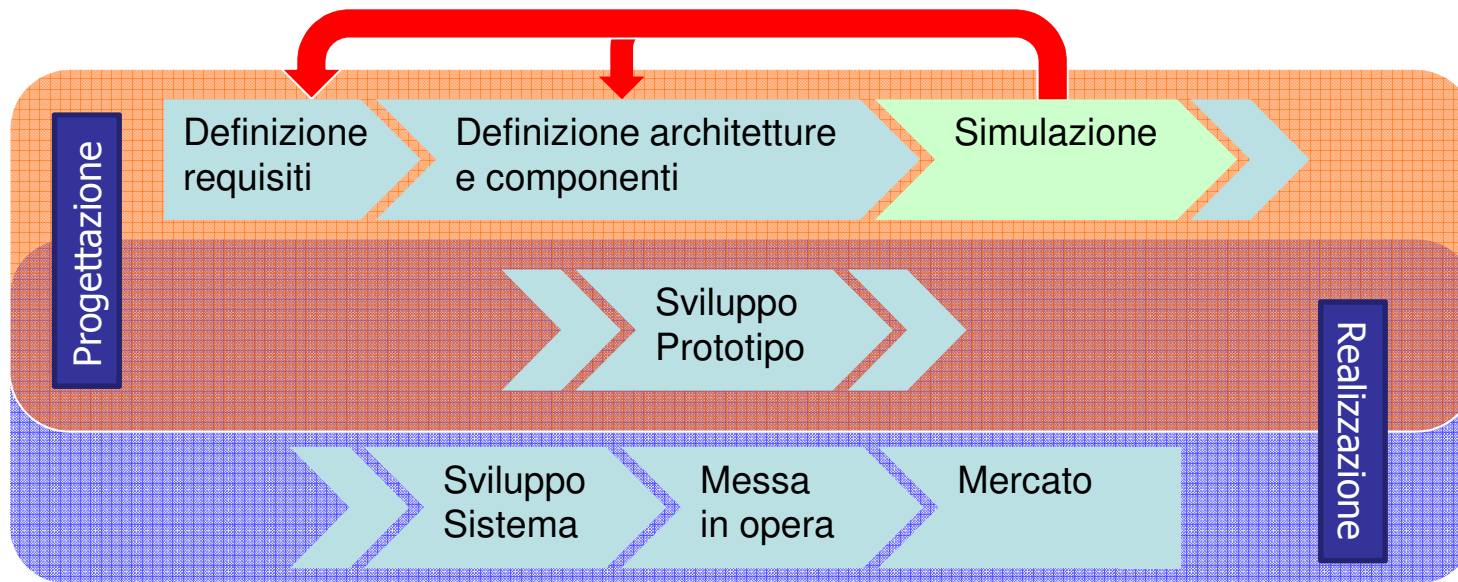


Simulazione ed Emulazione di sistemi satellitari



Premessa

- I sistemi ingegneristici (in particolare di telecomunicazioni ed in particolare satellitari) sono complessi
- Processo di sviluppo (a controreazione)



- Lo sviluppo (anche del prototipo) può essere molto costoso

Concetti sulla simulazione (1/2)

- Una simulazione è una attività di riproduzione di operazioni, di funzionalità o del comportamento di un sistema (o processo)
- Il sistema di riferimento della simulazione può essere:
 - Reale e già esistente
 - Evoluzione (desiderata) di un sistema esistente
 - Un sistema innovativo (progettazione)
- La simulazione si basa su un modello semplificato (anche se può essere molto complesso) del sistema di riferimento
- La simulazione avviene in un ambiente controllato ed gli esperimenti effettuati sono ripetibili
- Il simulatore deve essere validato confrontando i risultati con esperimenti condotti su sistemi veri

Concetti sulla simulazione (2/2)

- Scopo della simulazione:
 - Supportare l'attività di progettazione
 - Studiare l'evoluzione di un sistema (usando un tempo di simulazione anche differente dal tempo reale - accelerato/rallentato)
 - Valutare prestazioni e caratteristiche di soluzioni (anche alternative)
- Esito/Risultato della simulazione
 - Ottimizzazione di un'attività/sistema:
 - Riduzione dei costi (es. di progettazione e sviluppo)
 - Minimizzazione dei fallimenti ed identificazione delle criticità (es. rottura di un componente)
 - Ottimizzazione delle risorse (es. utilizzo di banda)
 - Velocizzazione del comportamento (es. tempo di risposta)
 - Identificazione della soluzione migliore a parità condizioni
 - Migliore comprensione del funzionamento ed isolamento delle cause ed effetti di un sistema

Aree di applicazione

- Ingegneria
 - micro/elettronica,
 - informatica,
 - civile,
 - telecomunicazioni,
 - gestionale
 - Sistemi di produzione
 - Sistemi di logistica
 - Mezzi di trasporto
 - Elaborazione dati
- Scienze naturali (fisica, chimica, geografia, astronomia, etc.)
- Ambiti socio/economico/politici
- Stime di funzionamento in ambienti pericolosi o per casi limite

Simulazione nelle telecomunicazioni (1/3)

- Nel caso di sistemi di telecomunicazione, di solito una simulazione mira a:
 - Ottenere una stima delle prestazioni di un sistema di comunicazione ad esempio in termini di velocità di trasmissione, qualità, affidabilità, etc.
 - Verificare l'efficacia di sistemi non ancora disponibili o in fase di progettazione
 - Supportare l'ottimizzazione di protocolli, applicazioni, procedure ancora in fase di sviluppo
 - Effettuare il dimensionamento del sistema prima di ottenere l'HW o il SW necessario (quante apparecchiature comprare, dove posizionarle, di che tipo, ecc.)

Simulazione nelle telecomunicazioni (2/3)

- In tal caso, una simulazione può non essere conveniente/efficace se:
 - il sistema da simulare esiste, ed il costo di sviluppo della simulazione è superiore al costo per realizzare un “testbed”;
 - La simulazione dà risultati ottenibili da semplici modelli matematici (carta e penna!);
 - La simulazione è basata su un modello troppo semplificato (quindi non rappresentativo) e produce risultati non validi oppure non interpretabili;
- È sempre un problema di compromesso (costo di sviluppo rispetto a risparmio di risorse)

Simulazione nelle telecomunicazioni (3/3)

- Un sistema di telecomunicazioni può essere composto da un numero talmente elevato (e costoso) di elementi e risorse coinvolte da rendere indispensabile una attività preliminare di simulazione (es. protocolli che lavorano su reti peer-2-peer di migliaia di nodi, pianificazione di copertura di una rete cellulare, etc.)
- Una simulazione può essere realizzata facendo leva su strumenti di simulazione generici già esistenti. I modelli di rete, traffico, etc. sono solitamente ben noti e di solito già disponibili su tali strumenti:
 - riduzione dei tempi e costi di sviluppo e validazione!
- Lo scopo di una attività simulativa può richiedere l'uso di diversi strumenti pre-esistenti:
 - Federazione di simulatori (integrazione)
- La peculiarità delle attività di simulazione richiesta può richiedere lo sviluppo di simulatori ad hoc
- Talvolta però i modelli disponibili oppure il dettaglio a cui si può arrivare sono limitati e i risultati devono essere consolidati con altri metodi:
 - Uso di HW vero - Emulazione

Federazione

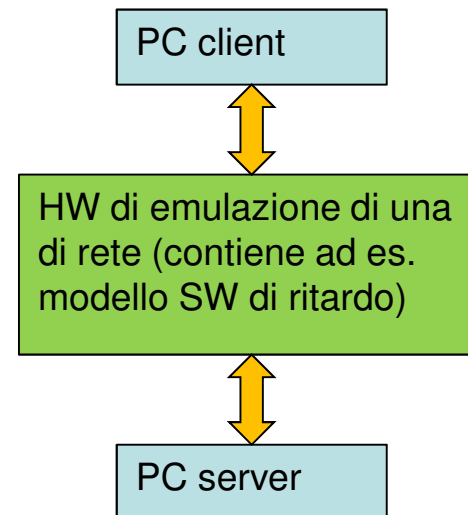
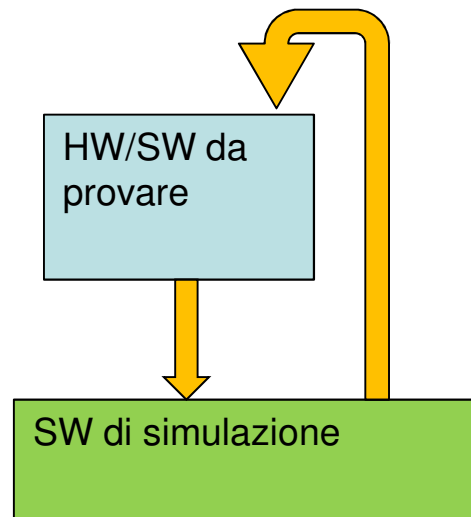
- Definire interfacce tra diversi strumenti di simulazione, in modo tale da usufruire delle funzioni necessarie in modo ottimale
 - Scambio di file di trace “offline”, serie temporali, tabelle, etc.
 - Comunicazione in tempo reale
- Particolarmente utilizzata da simulazioni CPU intensive, in cui le operazioni sono distribuite su più motori di simulazione paralleli
- Esistono standard di interfacciamento di simulatori:
 - High Level Architecture (HLA), standard IEEE1516
 - MPI/PVM (scambio di messaggi tipico di architetture cluster)
 - Soluzioni proprietarie specifiche
 - Soluzioni sviluppate ad-hoc (ad es. comunicazione su socket TCP/IP con file XML)

Emulazione (1/2)

- L'emulazione consiste nell'includere (parte) dell'HW/SW del sistema vero all'interno della Simulazione software
- La più diffusa è l'emulazione completa di un computer differente da quello ospitante. Un eseguibile può girare inalterato nell'emulatore software (es. emulatore di Game Boy, o una Macchina Virtuale)
- Per le telecomunicazioni per emulazione si intende tipicamente una simulazione che:
 - Interagisce in parte con HW vero (es. Hardware in the loop)
 - Include componenti HW che simulano solo alcune delle funzioni di un sistema complesso (es. emulatore di ritardo di processamento di un nodo di una rete)

Emulazione (2/2)

- Hardware in the Loop
- Emulatore di rete, testbed



La simulazione di sistemi satellitari (1/2)

- Nel caso specifico dei sistemi satellitari, la simulazione può essere a supporto di diverse attività, tra cui:
 - Ingegneria di sistema
 - Progettazione di satelliti (payload, struttura, antenne, peso, carburante, motori, pannelli solari, etc.)
 - Studio di missione (lancio del satellite e messa in orbita, struttura costellazioni, vita del satellite, etc.)
 - Progettazione e dimensionamento del segmento di terra (antenne, coperture, disponibilità)

La simulazione di sistemi satellitari (2/2)

➤ Telecomunicazioni:

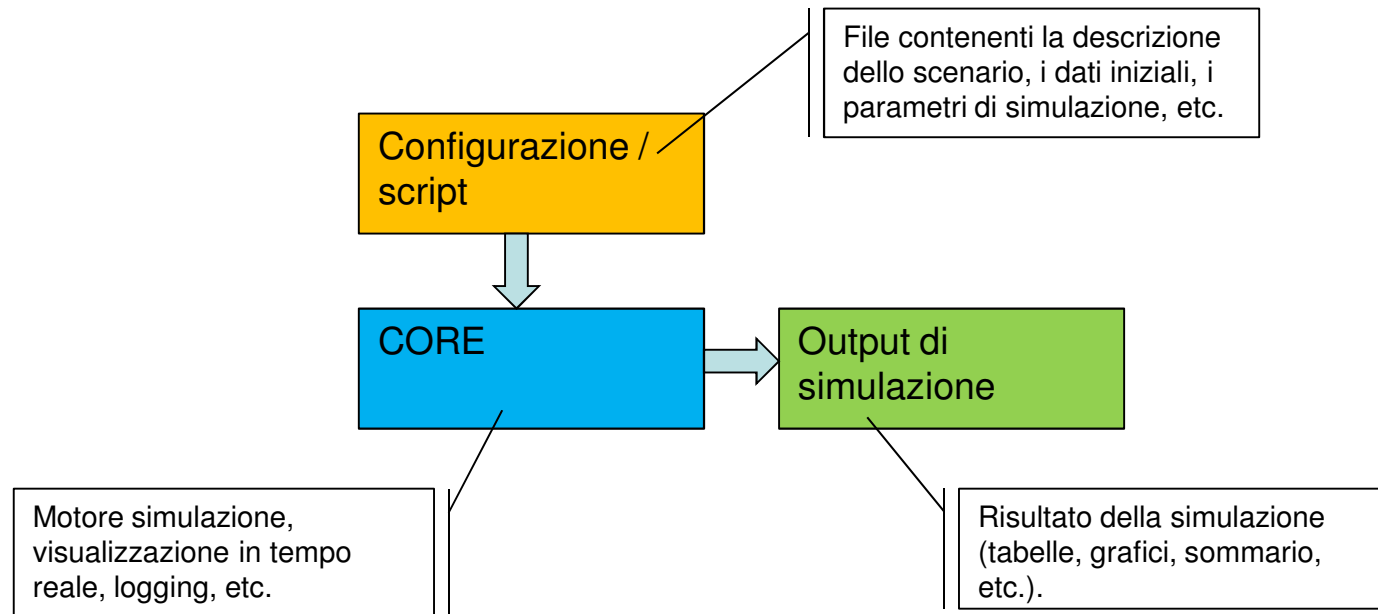
- Bilancio di radiocollegamento, velocità di trasmissione lorda, modulazione e codifica
- Studio di accesso al mezzo, (es. struttura dei frame TDMA, accesso random, etc.), efficienza di canale
- Studio comunicazione a pacchetto (studio 3), ottimizzazione di protocolli, compressione
- Studio della comunicazioni via satellite ed applicazioni (modelli di traffico, banda necessaria)
- Implicazioni dell'inclusione di segmenti satellitari su altre reti (reti ibride)
- Studio dell'handover tra tecnologie differenti, routing e multicast
- Analisi di traffico, carico di rete, aggregati di traffico
- Dimensionamento dei buffer, QoS (QoE), ritardi di rete

Ambienti di simulazione (1/4)

- Per costruire una simulazione di un sistema satellitare, si può ricorrere ad una serie di strumenti e ambienti di simulazione già predisposti, per ridurre i tempi di sviluppo:
 - Simulatori generici che offrono strutture dati, primitive, scambio messaggi, temporizzazione, visualizzazione grafica, etc. (da programmare in dettaglio);
 - Simulatori di rete a pacchetto (ma solitamente non comprendenti elementi satellitari), più “semplici”;
 - Simulatori dedicati al mondo satellitare (ma solitamente non adatti all’analisi di traffico/rete)
- Per casi particolari si deve ricorrere a simulatori realizzati da zero, o l’interscambio di dati tra simulatori differenti
- Può essere utile l’emulazione, come vedremo in seguito

Ambienti di simulazione (2/4)

- Simulatori tipicamente suddivisi in configurazione e “core” (nucleo di funzioni di base):



Ambienti di simulazione (3/4)

- Distinzione degli ambienti di Simulazione

	Vantaggi	Svantaggi
<u>Commerciale</u>	Semplicità di uso, interfaccia grafica, supporto, documentazione, aggiornamenti	Costo di licenza (può essere anche gratuito per funzionalità limitate, ma solitamente \$\$\$). Codice chiuso, funzioni aggiuntive non presenti disponibili a pagamento o su release future (forse)
<u>Open source</u>	Codice aperto, gratuito, sviluppato dalla comunità (aggiornamento allo stato dell'arte), possibilità di includere funzioni mancanti scrivendole	Documentazione e supporto su base volontaria, contributi al codice da entità diverse, parte grafica di solito limitata (file di testo da processare)

Ambienti di simulazione (4/4)

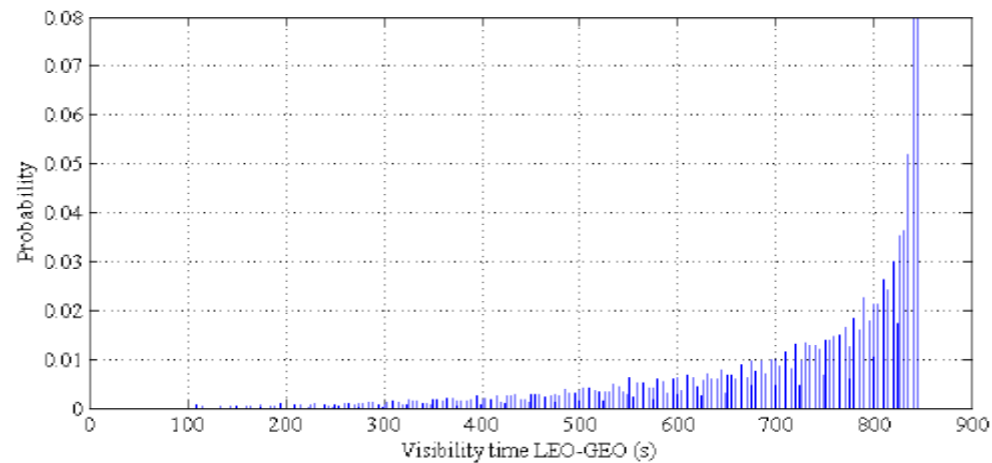
- Questo l'elenco dei tool di simulazione che verranno considerati nella seconda parte:
 - Commerciali:
 - MATLAB/Simulink
 - Satellite Tool Kit (STK)
 - Open
 - NS-2
 - NS-3
 - OMNeT++
- Ce ne sono numerosi altri! Qualche altro per completezza:
 - Opnet, Qualnet, Anylogic, J-Sim...
 - http://en.wikipedia.org/wiki/List_of_computer_simulation_software

Matlab (1/3)

- Noto in particolare per studi matematici, può essere utilizzato in modo molto flessibile includendo interfacce grafiche per l'ingresso dei parametri, la visualizzazione dei risultati etc.
- Consiste nella scrittura di file in linguaggio specifico (.m) che includono le chiamate a funzioni CORE, che vengono interpretati o compilati
- Estremamente potente, linguaggio di basso livello con programmazione complessa adatto in particolare al calcolo di funzioni complesse tra matrici.

Matlab (2/3)

- Può essere utilizzato per:
 - Analisi prestazioni dello strato fisico
 - Studio orbite, costellazioni e coperture
 - Input tramite file contenenti i parametri fisici
 - Risultati solitamente sotto forma di grafici/tabelle



Matlab (3/3)

- Interfaccia grafica utilizzata per studio di link budget

The screenshot shows the 'LinkBudgetTool' window with the title 'LINK BUDGET TOOL'. The interface is organized into several sections:

- HUB**: Includes 'Nome Sito' (COM-4_Planola), 'Longitudine (*)' (9.5), 'Latitudine (*)' (45.4), 'Altitudine (km)' (0.084), 'Elevazione (*)' (37.6), and 'EIRP (dBW)' (50).
- Satellite**: Includes 'Nome Satellite' (NSS 5), 'Payload' (Trasparente), 'Longitudine_sat (*)', 'Parametri RX' (Guadagno Ant. (dB): 60, T_antenna (K): 200, LNB figura rumore (dB): 25), and 'Parametri TX' (EIRP (dBW): 50).
- Terminale satellitare**: Includes 'Longitudine (*)' (9.5), 'Latitudine (*)' (45.4), 'Altitudine (km)' (0.084), 'Elevazione (*)' (37.6), 'Diametro Antenna (m)' (7), 'Efficienza downlink' (0.85), 'Polarizzazione (rad)' (0.7853981634), 'Disponibilità (%)' (99.5), and 'LNB figura rumore (dB)' (25.5).
- Perdite in uplink**: Includes 'Accoppiamento' (Perdita (dB): 0), 'Polarizzazione' (Perdita (dB): 1), 'Puntamento' (Perdita (dB): 1), and 'Propagazione' (Perdita (dB): 5).
- Parametri di sistema**: Divided into 'Uplink' and 'Down Link'.
 - Uplink**: 'Schema mod/cod' (QPSK 2/5), 'mod. order * code rate' (2), 'Fattore di Roll-Off' (0.35), 'Freq. uplink (GHz)' (25.5).
 - Down Link**: 'Schema mod/cod' (BPSK 1/2), 'mod. order * code rate' (BPSK 1/2, QPSK 1/4, QPSK 1/3, QPSK 2/5, QPSK 1/2, QPSK 3/5, QPSK 2/3, QPSK 3/4, QPSK 4/5, QPSK 5/6, QPSK 6/7, QPSK 8/9, QPSK 9/10, QPSK 17/18, 8PSK 1/2, 8PSK 3/5, 8PSK 2/3, 8PSK 3/4, 8PSK 5/6, 8PSK 7/8), 'Fattore di Roll-Off' (0.35), 'Freq. downlink (GHz)'.
- Parametri di uscita**: Includes 'Eb/N0 (dB)', 'C/N0 (dB/Hz)', 'C/N (dB)', and 'Bit rate (Mbit/s)'.
- Perdite in downlink**: Includes 'Accoppiamento' (Perdita (dB): 1), 'Polarizzazione' (Perdita (dB): 0), 'Puntamento' (Perdita (dB): 2), and 'Propagazione' (Temp. antenna (K), Attenuazione Tot (dB)).
- Parametri richiesti**: Includes 'Margine sistema (dB)' (5) and 'Eb/N0 richiesto (dB)' (5).

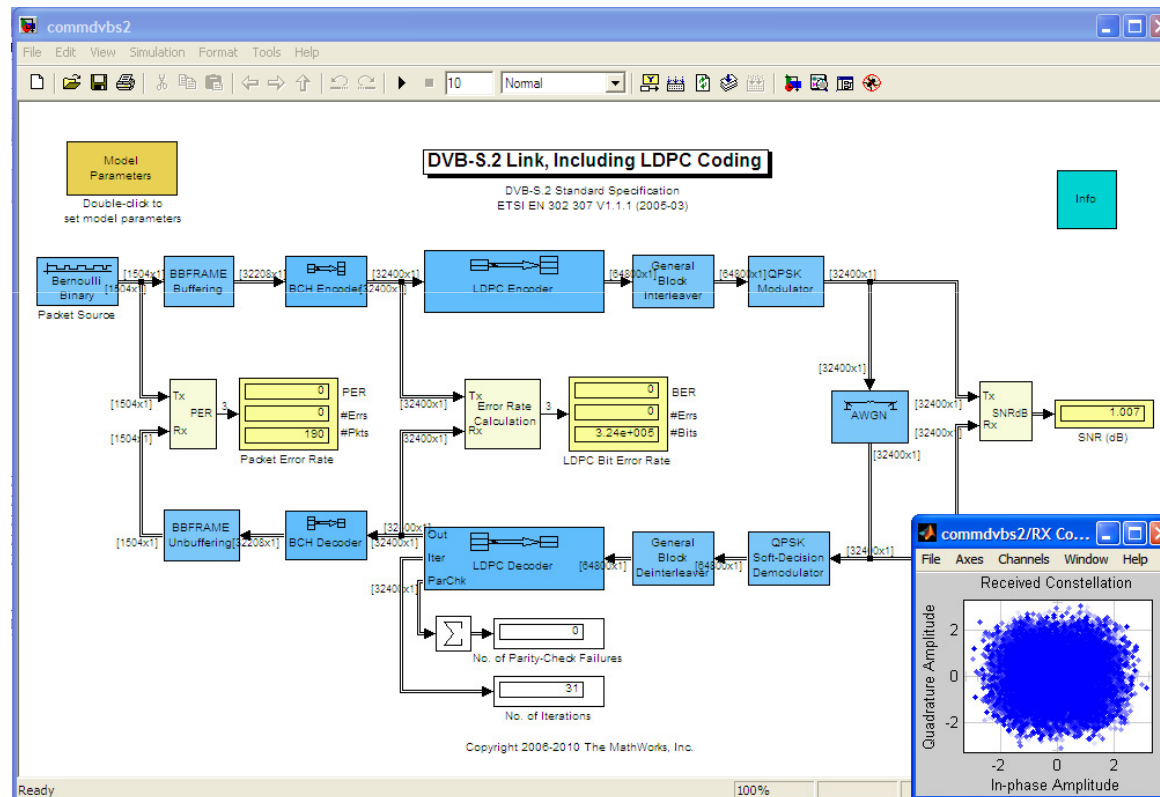
A 'Calcola' button is located at the bottom center of the interface.

Matlab/Simulink (1/2)

- Estensione di Matlab per definire modelli con blocchi logici interconnessi, concetto di tempo/eventi
- Composizione dello schema visuale (drag-and-drop)
- Proprietà dei blocchi facilmente editabili (schede presentate a video) e annidabili
- Si può usare ad esempio per:
 - Studio link budget, catene di ritrasmissione e perdite di canale
 - Scelta di modulazione e codifica per il dimensionamento di canale digitale

Matlab/Simulink (1/2)

- Catena TX-RX DVB-S2

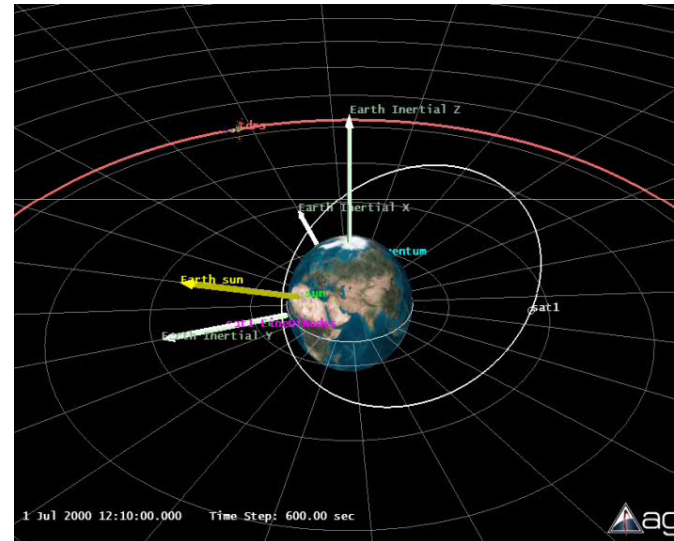


STK (1/3)

- Satellite Tool Kit - STK
 - Tool specifico per la simulazione di missioni spaziali, dai numerosi campi applicativi
 - Possibilità di richiamare dati esistenti (database di satelliti) ed aggiungerne nuovi
 - Estremamente potente nella visualizzazione grafica (3D) e utilizzo di modelli fisici dettagliati (gravitazione, orbite)
 - Supporta differenti pattern di antenne e aree di copertura
 - Calcolo evoluzione sistema in tempo reale
 - Comunicazione con simulatori esterni

STK (2/3)

- Esempio di simulazione di base con STK
- Visualizzazione Orbite
- Visualizzazione vettori di interesse
- Cambio riferimento per la visualizzazione
- Angolo di elevazione rispetto ad una stazione di terra
- Tutti i dati (angoli, tempi, etc.) sono anche disponibili in formato testuale



STK (3/3)

- Esempio di video professionale realizzato con STK
- In questo caso il risultato della simulazione è la ricostruzione di un "esperimento" Cinese realizzato nel 2007 di distruzione di un satellite:
 - > 2000 frammenti
 - Previsione di collisioni in futuro!

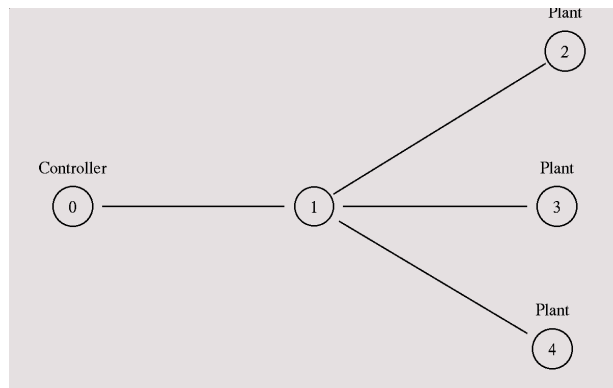


NS2 (1/3)

- Progetto Open source di un simulatore di rete ad eventi:
 - Aggiornato frequentemente (v2.35 Novembre 2011)
 - Mailing list utenti attiva (circa 200 mail al mese tra utenti e developers)
 - Standard de-facto per molti articoli scientifici sulle reti e TCP/IP
 - Molteplici contributi al codice da terze parti (“contrib”)
- Funzionalità di base in C++ (core):
 - Velocità di esecuzione
 - Object oriented, in pratica si compila una sola volta
- Linguaggio di scripting: TCL
 - Le simulazioni sono scritte in TCL (file di testo)
 - Non c’è bisogno di intervenire e ricompilare il core per eseguire diverse simulazioni
 - La complessità del core è nascosta all’utente finale

NS2 (2/3)

- Carenza di tool visuali per la creazione di topologie/scenari
 - Tool di conversione da altri formati (obsoleti)
 - Progetti in stallo
- Carenza di tool di visualizzazione dell'andamento della simulazione:
 - NAM lavora sui file generati a fine simulazione per ripercorrere cosa è successo graficamente:



NS2 (3/3)

- Le numerose estensioni (“contrib”) lo hanno reso uno standard per le simulazioni di rete ancora difficile da scalzare:
 - Soprattutto in ambito Wireless Networks/Mobility e routing:
http://nsnam.isi.edu/nsnam/index.php/Contributed_Code
- Possibilità di graficare le variabili di simulazione con un generatore di grafici integrato al progetto (xgraph) creando manualmente dei file con i valori di interesse.
 - In alternativa gnuplot, grace o anche excel.
- Supporto marginale all’emulazione con uno scheduler real-time sperimentale e senza il controllo sul traffico vero in transito
- Per ogni altro dettaglio fare riferimento alla documentazione completa:
<http://www.isi.edu/nsnam/ns/doc/index.html>

NS3 (1/3)

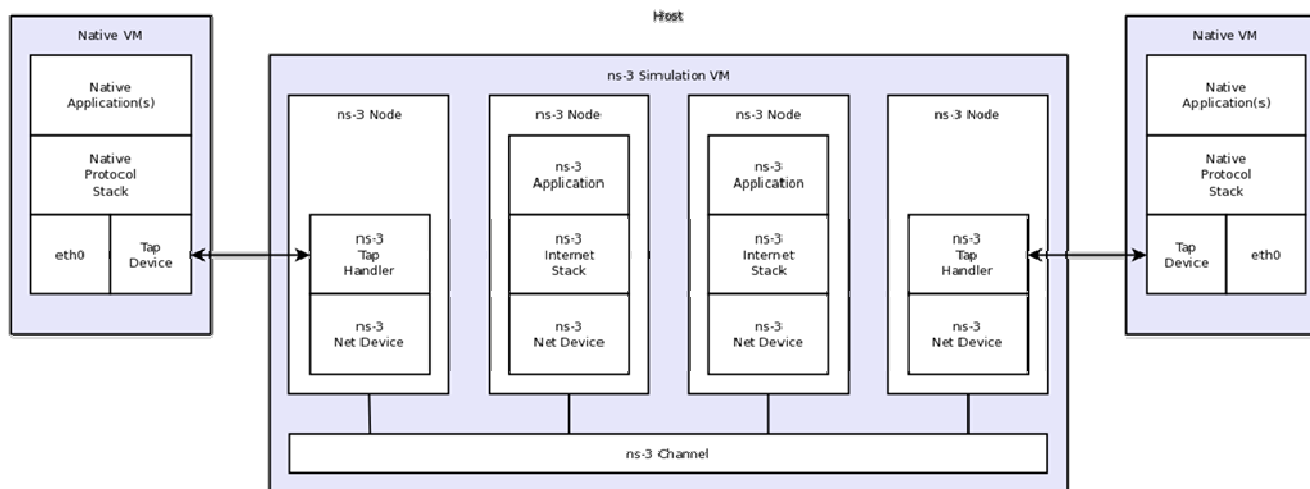
- Progetto volto a superare alcune delle limitazioni principali di NS2.
- Principali caratteristiche:
 - Nuovo linguaggio di scripting più intuitivo e potente: phyton (al posto del TCL)
 - Documentazione più rigorosa.
 - Traffico simulato compatibile con il formato di cattura pcap (traffico vero) ed analizzabile da Wireshark
 - Supporto avanzato all'emulazione, disponibili due modalità di integrazione con HW vero
 - Supporto alle macchine virtuali native
 - Protocolli utilizzati più vicini alle implementazione vere
 - Modifiche ai protocolli portabili su sistemi veri in maniera più semplice
 - Linguaggio del core C++ riscritto da zero e con design pattern di programmazione più avanzati e codice più controllato
 - Tracciamento degli eventi e gestione di output più flessibile

NS3 (2/3)

- Difetti evidenziati sin ora:
 - Progetto molto ambizioso, creato da zero con lo scopo di dare vita ad un simulatore molto potente non compatibile all'indietro (che risulta nella pratica molto complesso e di conseguenza di lento sviluppo, oltre a rendere non possibile il riuso di codice precedente)
 - Quantità di codice “contributed” da terze parti molto limitata
 - Requisiti di documentazione molto rigorosi (tempo di sviluppo di una nuova funzione < tempo per documentarla)
 - Non validato ed implicitamente riconosciuto come attendibile come NS2 (progetto relativamente giovane)

NS3 (3/3)

- Dettagli per il supporto all'emulazione:
 - Macchine virtuali con parti di simulazione NS3
 - TAPBridgeDevice: possibilità di associare ad un link tra due PC un tunnel terminato nel simulatore NS3 (il traffico vero sarà dirottato e trattato come traffico della simulazione)



Omnet++ (1/4)

- Insieme di librerie per la simulazioni ad eventi, orientato alle reti (ma non solo);
- Le funzionalità della simulazione richieste vengono incluse inserendo componenti già esistenti o introducendone nuovi (linguaggio basato su C++), per ottenere un vero e proprio eseguibile compilato
- I parametri di simulazione possono essere inseriti in uno (o più) file “ini”, permettendo di eseguire molteplici run con valori diversi senza ricompilare il tutto
- Integrato dentro l' IDE Eclipse, ereditandone tutte le caratteristiche ed introducendone altre (ad es. visualizzazione grafica degli scenari)
- Può realizzare emulazione con la connessione su socket veri che terminano nel modello simulato (server HTTP!)

Stop, pause,
step...

Omnet++ (2/4)

The screenshot displays the OMNeT++ ClientServer simulation interface. The top menu bar includes File, Edit, Simulate, Trace, Inspect, View, Options, and Help. Below the menu is a toolbar with icons for running, pausing, and stepping through the simulation. The main window is divided into three panes:

- Timeline:** A horizontal axis at the top showing time in seconds (0.001 to 100). It includes markers for events like 'timeout-3', 'timeout-5', 'timeout-6', 'timeout-7', 'timeout-8', 'timeout-9', and 'timeout-10'. A label 'Timeline' points to this pane.
- Log:** A central pane displaying a list of events with timestamps and descriptions. For example, 'Event #25 T=8.145411712323 ClientServer.client[6] (Client, id=10), on 'DYNA_CONN_ACK' (DynaPacket, id=9) got DYNA_CONN_ACK, my server process is ID=12'. A label 'Log' points to this pane.
- Prossimi eventi (Upcoming events):** A list on the left side showing upcoming events, such as 'timeout-3 (cMessage) (id=0)', 'DATA(result) (DynaDataPacket) (id=14)', and various 'timeout-x' events. A label 'Prossimi eventi' points to this list.

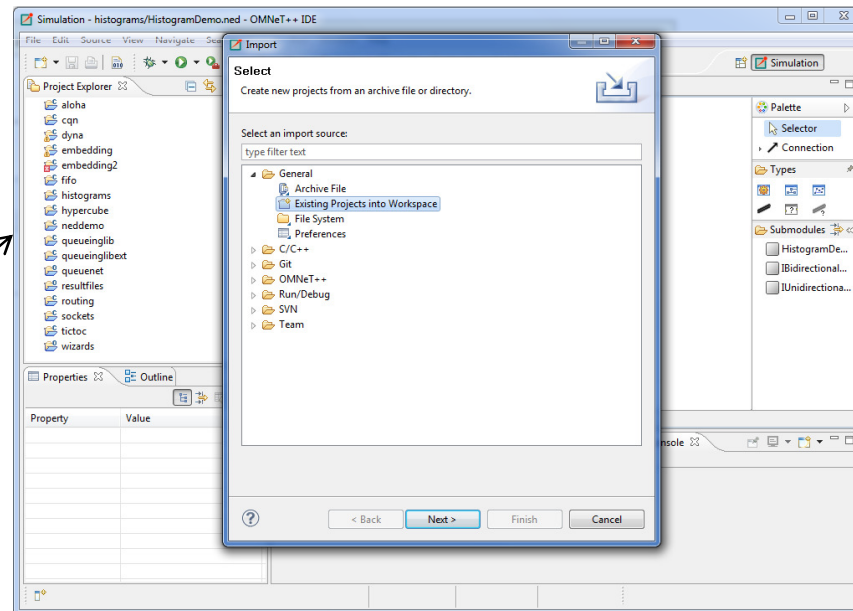
Below the main window is a smaller window titled '(ClientServer) ClientServer' showing a network diagram. The diagram illustrates a central 'server' connected to a 'switch' (labeled 'q0'), which is then connected to a series of clients labeled 'client[0]' through 'client[7]'. A label 'Animazione grafica in tempo reale' points to this diagram.

33

Omnet++ (3/4)

- Installazione
 - Seguire la guida su:
<http://www.omnetpp.org/doc/omnetpp41/InstallGuide.pdf>
 - Fare click su Menu File → Import e selezionare tramite exsisting projects la cartella “samples” dell’ installazione
 - Navigare nelle cartelle, selezionare i file .ned e lanciare (tasto run) da Eclipse. Ad esempio *tictoc/tictoc1.ned*

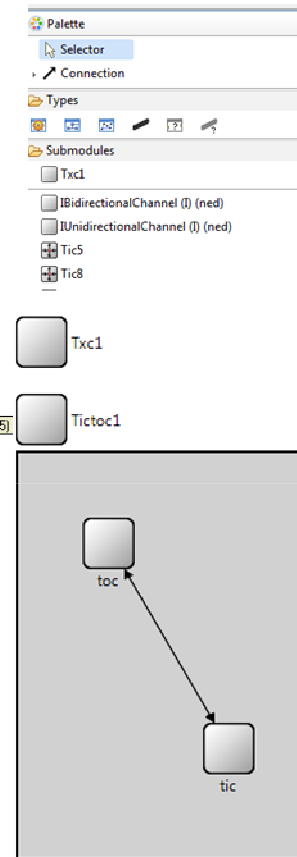
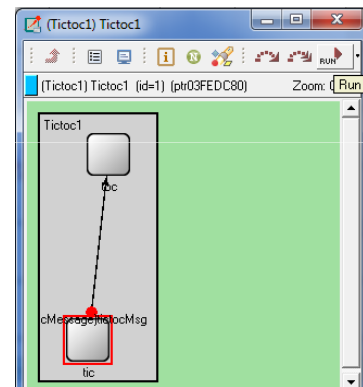
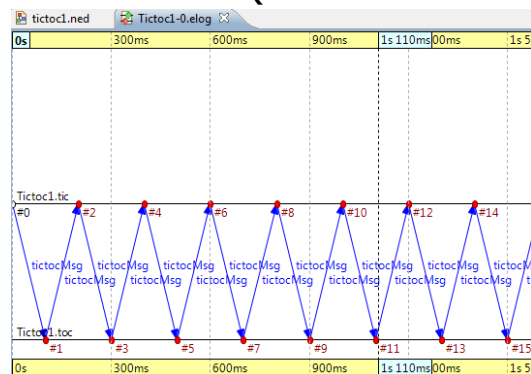
Eclipse IDE!



Omnet++ (4/4)

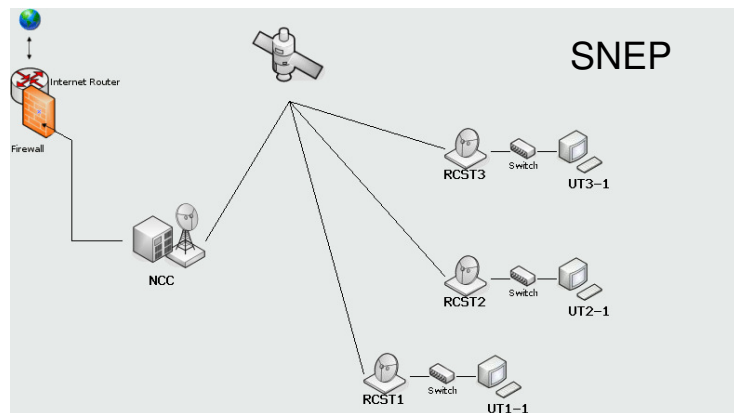
- Omnet++: Esecuzione dell' esempio tictoc
 - Scambio di messaggi tra due nodi
- Editor visuale (file .ned)!
- Console di esecuzione animata

- Sequence Diagram della Simulazione (abilitare eventlog)



Emulatore di reti satellitari (1/3)

- Per riprodurre le caratteristiche di una rete satellitare con traffico vero (ed in tempo reale) è possibile utilizzare una rete di macchine basate su Sistema Operativo Linux
- Ogni macchina ricoprirà le funzionalità tipiche di una rete satellitare: Gateway di accesso ad internet, NCC, satellite, modem satellitari e terminali utente
- E' stata in particolare progettata per riprodurre un sistema conforme allo standard DVB-RCS, chiamata SNEP

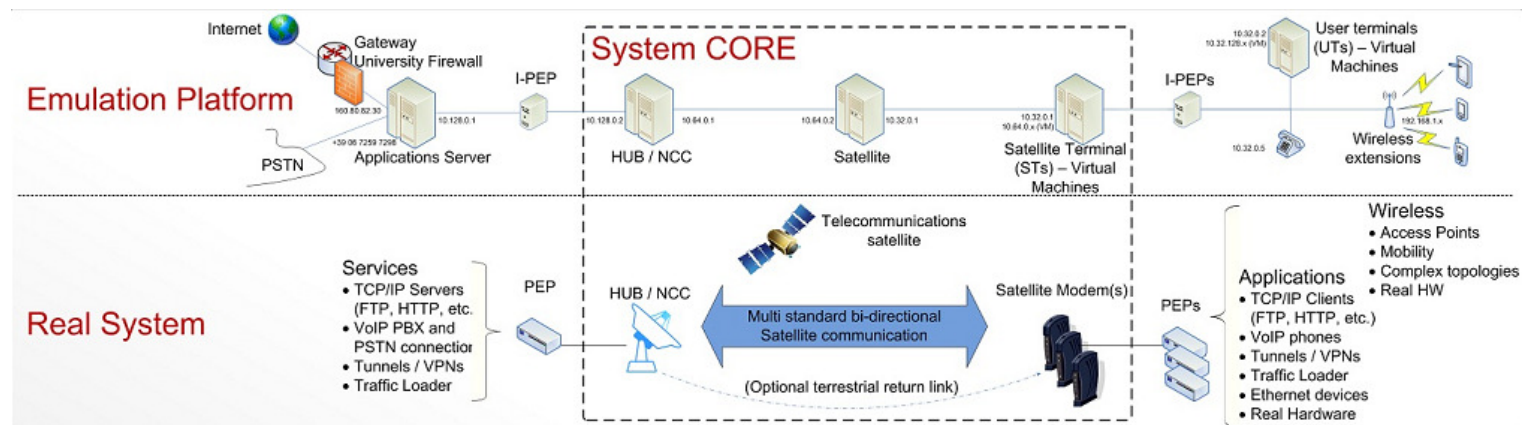


Emulatore di reti satellitari (2/3)

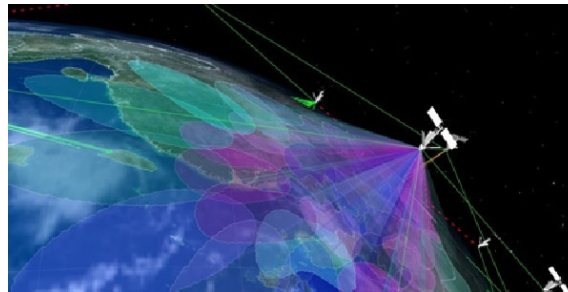
- Lo standard DVB-RCS
 - Prevede la trasmissione verso tutti i terminali satellitari utilizzando lo standard di broadcast DVB-S (S2)
 - Prevede un canale di ritorno condiviso con una tecnica di accesso TDMA (Multi frequenza) e richiesta esplicita di risorse (meccanismo di banda su domanda di tipo DAMA)
 - Prevede funzioni di accesso, autenticazione (Logon), e sincronizzazione per la trasmissione nella trama, funzioni di richiesta di banda ed assegnazione di risorse (Burst Time Plan)
- Tutto ciò è stato riprodotto (emulato) nello SNEP, per mezzo di eseguibili distribuiti ed il controllo dei pacchetti in transito su ciascun nodo. Lo SNEP prevede anche l'uso di macchine virtuale per modulare il numero di nodi necessari all'emulazione.

Emulatore di reti satellitari (3/3)

- Possono essere utilizzate applicazioni vere ed hardware addizionale per testare il comportamento su una rete satellitare
- Integrabile con altri simulatori
- Nella seconda parte si effettuerà una demo sull'utilizzo dell'emulatore.

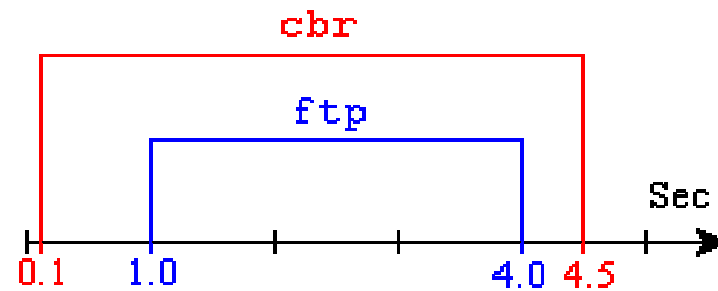
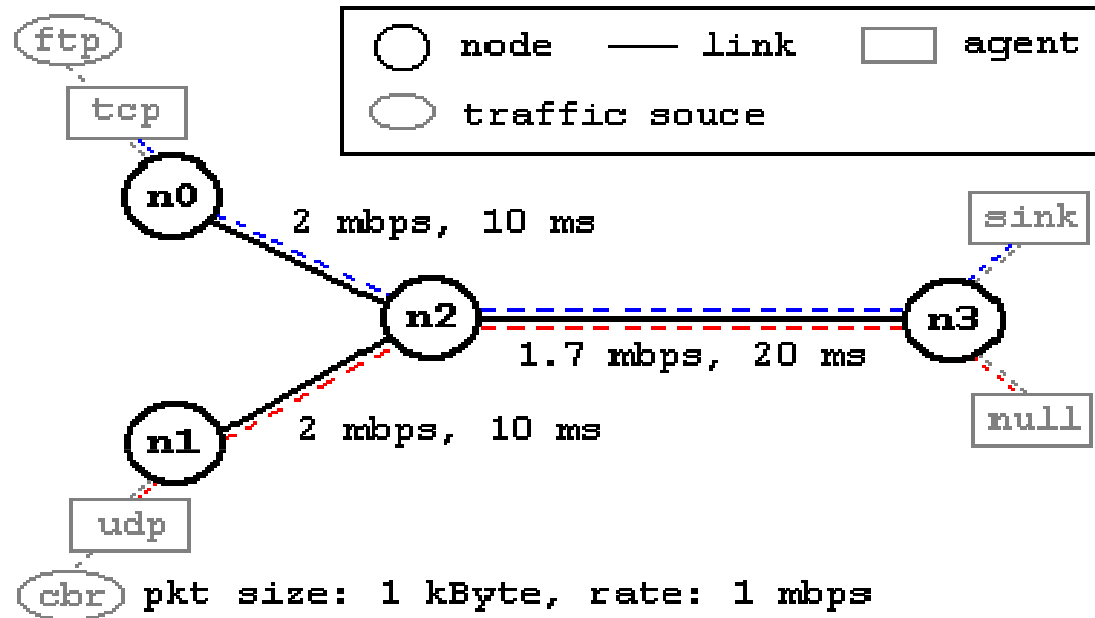


Esempi pratici di simulazione NS2 ed Emulazione Linux



NS2

- Si compila dai sorgenti su Linux/Unix(MAC) in modo semplice;
- Può essere compilato su Windows (più complicato!!). Eseguibili già pronti disponibili;
- Primo test di traffico su topologia semplice
- Esecuzione di NAM, il visualizzatore di simulazione
- Migliorie allo script di test
- Simulazione NS2 in ambiente satellitare
- Script preso in parte da “NS2 by Example” su web



TCL

- Non serve definire le variabili, nè il tipo
- Assegnazione: $A = 5$ si scrive `set A 5`
- Vettori $V=(1;5)$ si scrive `set V(0) 1; set V(1) 5`
- Assegnazione stringa `set str "Hello world!"`
- Utilizzare la variabile con il `$` (`puts` è la stampa a schermo) `puts $str`
- Operazioni matematiche ed annidamento di funzioni:
 $B=A*5$ si scrive `set B [expr $A * 5]`
- Condizioni: `if { $B==5 } { ... } else { ... }`
- Ciclo di operazioni "for": `for {set i 0} {$i < $MAX} {incr i} { ...Operazioni... }`
- Definizione funzioni:
`proc funzione { param1 ... param n } {}`

Script di simulazione 1/11

```
#Crea un oggetto "Simulazione"
set ns [new Simulator]; # Simulator fa riferimento
                        # al core di NS2 in C++

#Apri un file ed abilita i log per NAM
set logfile [open "out.nam" w]
$ns namtrace-all $logfile

#Crea 4 nodi
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
```

Script di simulazione 2/11

```
#Crea links tra i nodi
```

```
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
```

```
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
```

```
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail
```

```
#Imposta dimensione del buffer sul link (n2<->n3)
```

```
$ns queue-limit $n2 $n3 10
```

Script di simulazione 3/11

```
#Setup di una connessione TCP
set tcp [new Agent/TCP] ;#TCP è un "agent"
                        # corrispondete ad una
                        # classe C++ core con
                        # variabili "bindate"

$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink

$ns connect $tcp $sink
$tcp set fid_ 1 ;# FLOW identifier..."bind"
```

Script di simulazione 4/11

```
#Associa una applicaz. FTP alla connessione TCP
set ftp [new Application/FTP]

$ftp attach-agent $tcp ;# !! Diverso dall'
                                # $ns attach-agent

$ftp set type_ FTP
```

Script di simulazione 5/11

```
#Setup di una connessione UDP
```

```
set udp [new Agent/UDP]
```

```
$ns attach-agent $n1 $udp
```

```
set null [new Agent/Null]
```

```
$ns attach-agent $n3 $null
```

```
$ns connect $udp $null
```

```
$udp set fid_ 2
```

Script di simulazione 6/11

```
#Setup di un flusso CBR su UDP  
set cbr [new Application/Traffic/CBR]  
$cbr attach-agent $udp  
  
$cbr set type_ CBR  
$cbr set packet_size_ 1000  
$cbr set rate_ 1mb  
$cbr set random_ false
```

Script di simulazione 7/11

```
#Fine della configurazione Topologia/Traffico

#Definizione di una procedura di fine
proc finish {} {
    global ns logfile
    $ns flush-trace
    close $logfile; #Chiudi il file NAM di log
    #Esegue NAM sul file e ritorna:
    exec nam out.nam &
    exit 0
}
```

Script di simulazione 8/11

```
#Definisci eventi temporizzati
```

```
$ns at 0.1 "$cbr start"
```

```
$ns at 1.0 "$ftp start"
```

```
$ns at 4.0 "$ftp stop"
```

```
$ns at 4.5 "$cbr stop"
```

```
#Esegue la procedura finish al termine di tutto
```

```
$ns at 5.0 "finish" ; # La finish eseguirà anche NAM
```

```
#Quando tutto è definito, alla fine del file:
```

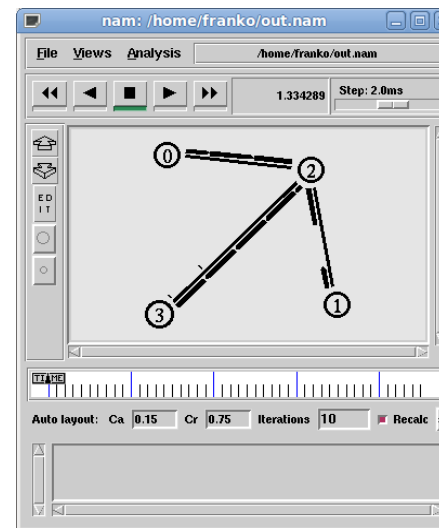
```
$ns run
```

Esecuzione e output

1) Su console:

```
$/NS2/:~$ ns ns-simple.tcl
```

2) Visualizzazione grafica:



Script di simulazione 9/11

```
#Aggiunta di altri dettagli per l'output grafico.
```

```
#Colori flussi (definiti prima con "fid_")
```

```
$ns color 1 Blue
```

```
$ns color 2 Red
```

```
#Posizione relativa dei nodi
```

```
$ns duplex-link-op $n0 $n2 orient right-down
```

```
$ns duplex-link-op $n1 $n2 orient right-up
```

```
$ns duplex-link-op $n2 $n3 orient right
```

```
#E della strozzatura (buffer)
```

```
$ns duplex-link-op $n2 $n3 queuePos 0.5
```

Script di simulazione 10/11

```
#Aggiunta di Output testuale
```

```
#Scrivi caratteristiche traffico CBR
```

```
puts "CBR packet size = [$cbr set packet_size_]"
```

```
puts "CBR interval = [$cbr set interval_]" ; # set  
qui significa get (!)
```

Script di simulazione 11/11

```
#Stampa variabile dinamica

proc finish {} {
    global ns logfile tcp sink
    puts "Bytes RX TCP = [$sink set bytes_]"
    $ns flush-trace
    close $logfile; #Close the NAM trace file
    #Execute NAM on the trace file:
    exec nam out.nam &
    exit 0
}
```

Output simulazione

Su console:

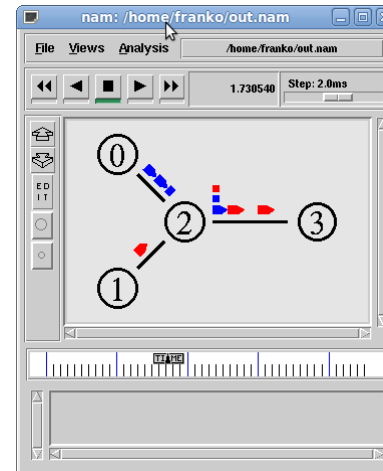
```
$/NS2/:~$ ns ns-simple_record.tcl
```

```
CBR packet size = 1000
```

```
CBR interval = 0.008000000000000000000002
```

```
Bytes RX TCP = 228840
```

2) Grafico:



Modifica allo script:log

Inserire il comando per tracciare su file tutti i pacchetti che transitano nella rete (log completo):

+ : Pacchetto generato a livello 4

- : Pacchetto inviato allo strato fisico

r : Pacchetto ricevuto dal destinatario

```
set log_all [open "nstraceall.dat" w]
```

```
$ns trace-all $log_all
```

+	1.060696	0	2	tcp	1040	-----	1	0.0	3.0	1	2
-	1.060696	0	2	tcp	1040	-----	1	0.0	3.0	1	2
+	1.060696	0	2	tcp	1040	-----	1	0.0	3.0	2	3
-	1.064856	0	2	tcp	1040	-----	1	0.0	3.0	2	3
r	1.074856	0	2	tcp	1040	-----	1	0.0	3.0	1	2
+	1.074856	2	3	tcp	1040	-----	1	0.0	3.0	1	2

Ulteriore modifica: grafici

Definire una semplice funzione di record, che “si chiama” periodicamente per poter generare un file di output:

```
proc record {} {  
    global ;#“elenco variabili globali di interesse”  
    set now [$ns now] ;# Tempo di simul. in secondi  
    set delta 0.1  
  
    ...operazioni...  
  
    $ns at [expr $now + $delta] "record"  
}
```

Script di simulazione con “record” 1/3

```
#Variabili globali:
set plotfile(0) [open "cwnd.log" w]
set plotfile(1) [open "rtt.log" w]
#Funzione record
proc record {} {
    global ns plotfile tcp sink
    set now [$ns now] ;# Tempo di simul. in secondi
    set delta 0.1
    puts $plotfile(0) "$now [$tcp set cwnd_];#formato x y
    puts $plotfile(1) "$now [$tcp set srtt_]"
    $ns at [expr $now + $delta] "record"
}
```

Script di simulazione con “record” 2/3

```
#Aggiornamento della finish per chiudere i files aperti:
proc finish {} {
    global ns logfile plotfile tcp sink
...
    close $logfile; #Chiudi il file NAM di log
    close $plotfile(0)
    close $plotfile(1)
...
    exit 0
}
...
$ns at 0.0 "record" ;# viene lanciata la prima volta
insieme agli altri elementi temporizzati
```

Script di simulazione con “record” 3/3

File risultante:

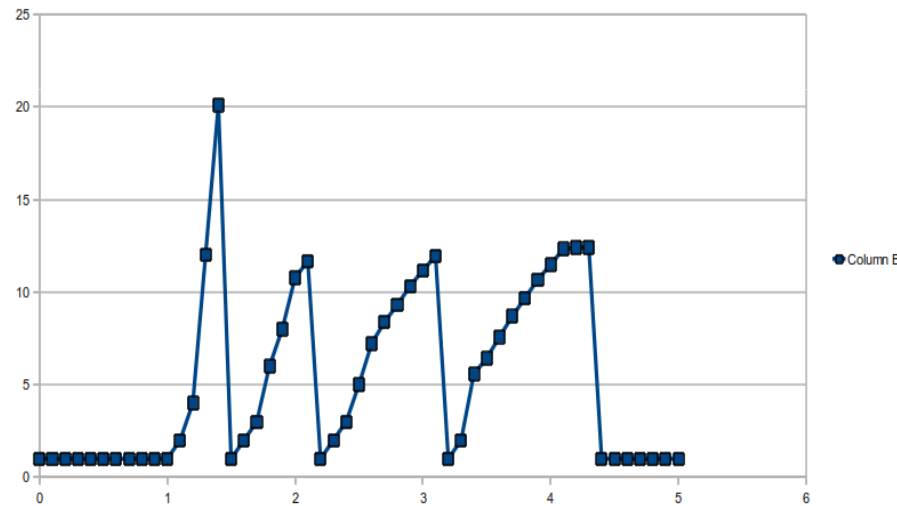
“cwnd.agr”

```

...
2.00000000000000000004 60
2.10000000000000000005 63
2.20000000000000000006 63
2.30000000000000000007 63
2.40000000000000000008 63
2.50000000000000000009 65
2.60000000000000000001 64
2.70000000000000000011 64
2.80000000000000000012 63
2.90000000000000000012 66
3.00000000000000000013 66
3.10000000000000000014 69
3.20000000000000000015 69
3.30000000000000000016 69
3.40000000000000000017 67
3.50000000000000000018 67
...

```

Graficabile con: Excel (dispersione), gnuplot, grace, xplot (parte di NS2)



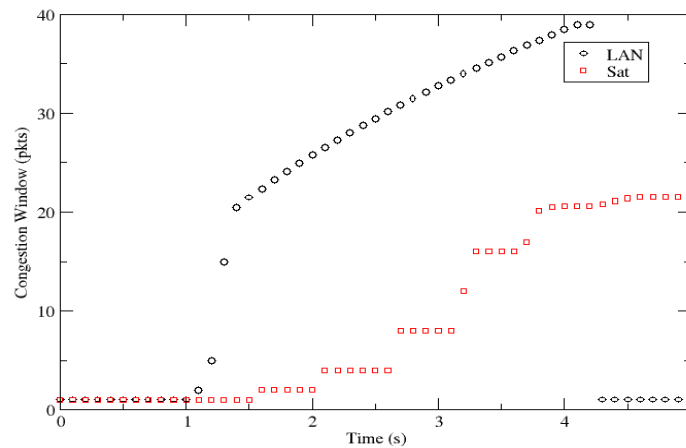
Openoffice (import – XY scatter plot)

Simulazione in NS2: ambiente satellitare (1/2)

É possibile introdurre ritardi maggiori sui singoli link per simulare un semplice collegamento punto-punto satellitare:

```
$ns duplex-link $n2 $n3 512kb 250ms DropTail
```

Già grande impatto sul protocollo TCP:



Simulazione in NS2: ambiente satellitare (2/2)

- È altresì possibile utilizzare le funzionalità del package “satellite” di NS2, ma
 - Compatibilità in avanti di codice NS1 (alcune incompatibilità, ad esempio con modelli di errore), con conseguente creazione di link in maniera differente dal modello NS2 standard;
 - Disponibilità limitate di strati “Link” e “MAC” che rispecchino tecnologie attuali;
 - Topologia e routing indipendente dal resto della rete (nodo “geo-repeater” e comando “compute_routes” solo per la sottorete satellitare);
 - ✓ Calcolo dei ritardi in base alla posizione geografica dei nodi.

Esempio creazione nodo satellitare

```
# Configure the node generator for NCC
$ns node-config -satNodeType terminal \
                -llType LL/Sat \
                -ifqType Queue/DropTail\
                -ifqLen 720 \
                -macType Mac/Sat \
                -phyType Phy/Sat \
                -channelType Channel/Sat \
                -downlinkBW 1Mbit \
                -wiredRouting ON

# Create one NCC terminal
set $long 18.0
set $lat 93.0
set ncc [$ns node]
$ncc set-position $long $lat
```

Satellite in NS2: elementi disponibili (1/4)

- LL/Sat → Link layer.
 - Si occupa di trovare il nodo destinatario per la comunicazione
 - Inserisce il numero di sequenza (MAC)
 - Inserisce il protocol type ad Ethernet (!)
 - Esempio di commenti nel codice:

```
/*In the satellite case, we don't arp (for now).  
* If someone wants to add arp,  
* look at how the wireless code does it.*/
```

Satellite in NS2: elementi disponibili (2/4)

- Mac/Sat → MAC layer.
 - Si comporta come Slotted Aloha. Determina collisioni su canale condiviso e le registra (scartando i pacchetti coinvolti).
- Ha un'altra sottoclasse:
 - Mac/Sat/UnslottedAloha

Satellite in NS2: elementi disponibili (3/4)

- Phy/Sat → Physical layer.
 - Inoltra i pacchetti tenendo in considerazione banda e ritardo del canale
 - Esempio di commenti nel codice riferiti alla chiamata sendup (vai allo strato superiore (MAC)):

```
// Note that this doesn't do that much right now.  
// If you want to incorporate  
  
// an error model, you could insert a  
// "propagation" object like in the  
  
// wireless case.
```

Satellite in NS2: elementi disponibili (4/4)

- Channel/Sat → Canale di propagazione.
 - Crea le interfacce logiche tra PHY ed il canale fisico
 - Calcola il ritardo di propagazione tra le stazioni coinvolte
- Logging di sistema modificato, per tenere conto delle informazioni aggiuntionali dei nodi satellitari:

```
+ 0.4942 1 2 tcp 1500 ----- 1 1.0 2.0 2 3 12.00 43.00 16.00 48.00
```

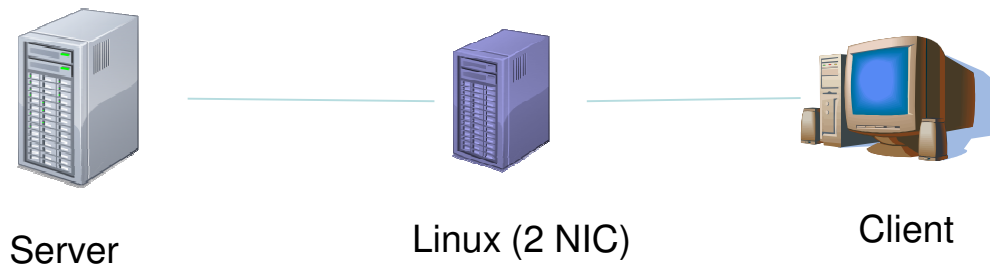
Satellite in NS2: elementi aggiunti

- Nell'ambito delle nostre attività di ricerca, abbiamo sviluppato una modifica a Mac/Sat (chiamata Mac/Sat/Dama) per riprodurre in modo fedele le dinamiche di un sistema DVB-RCS, compresi profili di richieste di capacità su base domanda per i terminali satellitari e diversi algoritmi di allocazione:

<http://www.tlcsat.uniroma2.it/DAMA>

Emulazione

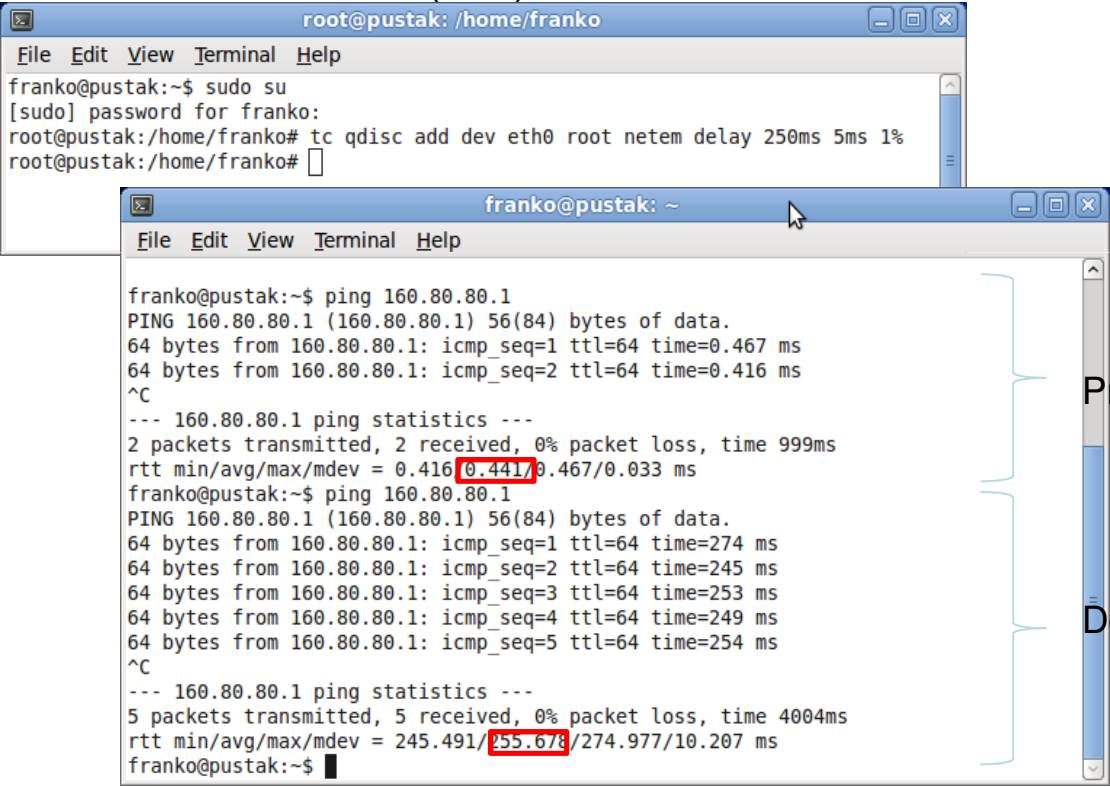
- Esempio di semplice emulatore di ritardo di rete tramite un Gateway/PC Linux
 - PC con due interfacce di rete:



```
root@pustak: /home/franko
File Edit View Terminal Help
root@pustak:/home/franko# cat /proc/sys/net/ipv4/ip_forward
0
root@pustak:/home/franko# echo 1 > /proc/sys/net/ipv4/ip_forward
root@pustak:/home/franko# cat /proc/sys/net/ipv4/ip_forward
1
root@pustak:/home/franko#
```

Emulazione

- Inserimento ritardo di propagazione su di una delle schede di rete (NIC):



The image shows two terminal windows. The top window, titled 'root@pustak: /home/franko', shows the execution of 'sudo su' and the configuration of a network delay on the 'eth0' interface using the command 'tc qdisc add dev eth0 root netem delay 250ms 5ms 1%'. The bottom window, titled 'franko@pustak: ~', shows two ping tests to 160.80.80.1. The first test, labeled 'Prima' with a bracket, shows low latency (0.467 ms and 0.416 ms). The second test, labeled 'Dopo' with a bracket, shows significantly higher latency (274 ms to 254 ms) after the delay was applied. In both tests, the 'rtt min/avg/max/mdev' line shows an average delay that has increased from 0.441 ms to 255.678 ms.

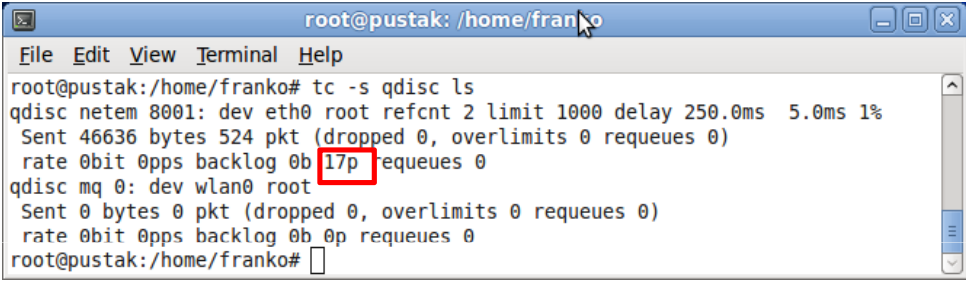
```
root@pustak: /home/franko
File Edit View Terminal Help
franko@pustak:~$ sudo su
[sudo] password for franko:
root@pustak:/home/franko# tc qdisc add dev eth0 root netem delay 250ms 5ms 1%
root@pustak:/home/franko#

franko@pustak: ~
File Edit View Terminal Help

franko@pustak:~$ ping 160.80.80.1
PING 160.80.80.1 (160.80.80.1) 56(84) bytes of data.
64 bytes from 160.80.80.1: icmp_seq=1 ttl=64 time=0.467 ms
64 bytes from 160.80.80.1: icmp_seq=2 ttl=64 time=0.416 ms
^C
--- 160.80.80.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.416/0.441/0.467/0.033 ms
franko@pustak:~$ ping 160.80.80.1
PING 160.80.80.1 (160.80.80.1) 56(84) bytes of data.
64 bytes from 160.80.80.1: icmp_seq=1 ttl=64 time=274 ms
64 bytes from 160.80.80.1: icmp_seq=2 ttl=64 time=245 ms
64 bytes from 160.80.80.1: icmp_seq=3 ttl=64 time=253 ms
64 bytes from 160.80.80.1: icmp_seq=4 ttl=64 time=249 ms
64 bytes from 160.80.80.1: icmp_seq=5 ttl=64 time=254 ms
^C
--- 160.80.80.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 245.491/255.678/274.977/10.207 ms
franko@pustak:~$
```

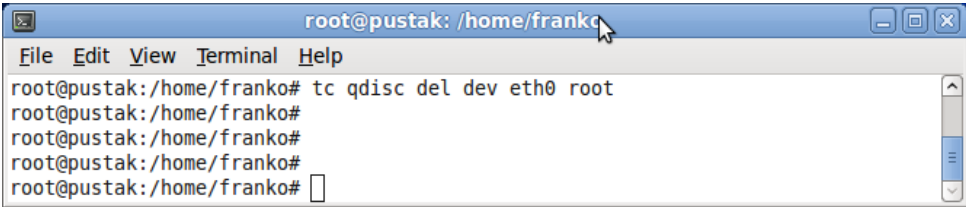
Emulazione

- Statistiche:



```
root@pustak: /home/franko
File Edit View Terminal Help
root@pustak:/home/franko# tc -s qdisc ls
qdisc netem 8001: dev eth0 root refcnt 2 limit 1000 delay 250.0ms 5.0ms 1%
Sent 46636 bytes 524 pkt (dropped 0, overlimits 0 requeues 0)
rate 0bit 0pps backlog 0b 17p requeues 0
qdisc mq 0: dev wlan0 root
Sent 0 bytes 0 pkt (dropped 0, overlimits 0 requeues 0)
rate 0bit 0pps backlog 0b 0p requeues 0
root@pustak:/home/franko#
```

- Ritorno allo stato iniziale:

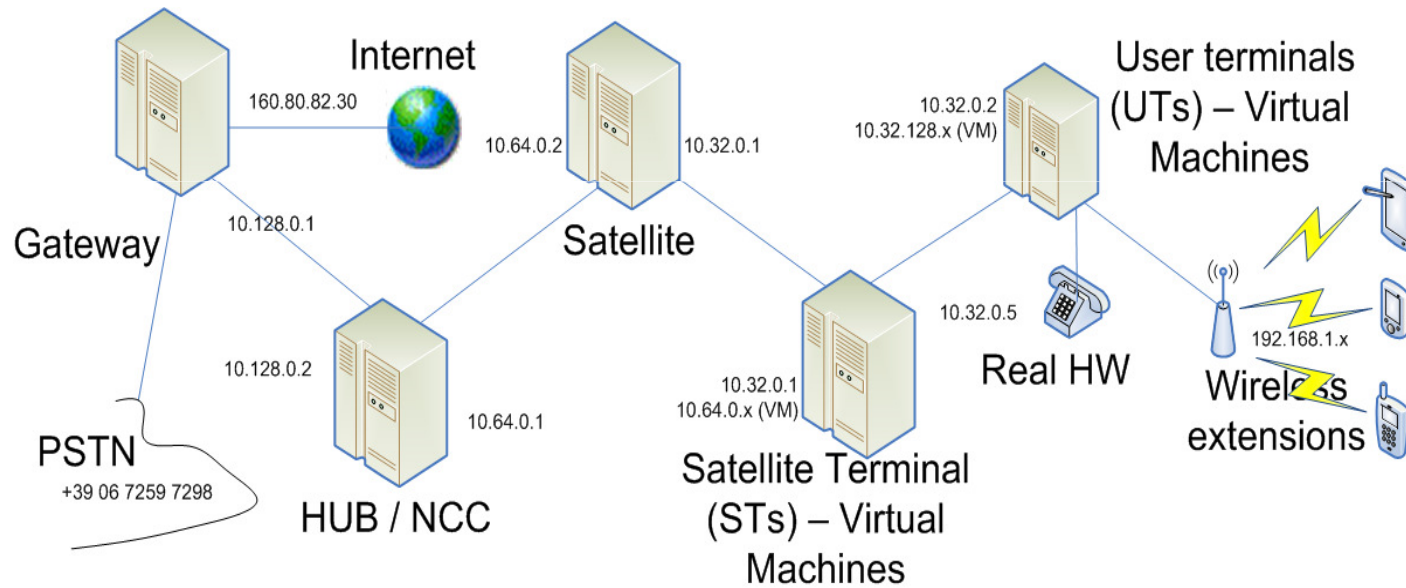


```
root@pustak: /home/franko
File Edit View Terminal Help
root@pustak:/home/franko# tc qdisc del dev eth0 root
root@pustak:/home/franko#
root@pustak:/home/franko#
root@pustak:/home/franko#
root@pustak:/home/franko#
```

Emulatore SNEP

- Nell'ambito delle nostre attività di ricerca, abbiamo sviluppato una piattaforma completa di emulazione per sistemi Satellitari bidirezionali (DVB-RCS), multiterminale
- Diverse macchine Linux per ricreare il comportamento caratteristico dei nodi di una rete satellitare (es. ritardo di propagazione applicato da un router chiamato SAT)
 - Più complesso dell'esempio precedente: diversi ritardi a seconda della posizione geografica del terminale, diversi modelli di errore, strozzature di banda dinamiche, segnalazione, QoS....etc.
 - Interfacciamento con altri testbed (estensione con altre tecnologie/HW o simulatori esterni)

Emulatore SNEP



Test Online

Test utilizzo emulatore online

Grazie per l'attenzione!

Francesco Zampognaro

Contatti:

zampognaro@ing.uniroma2.it

<http://www.tclsat.uniroma2.it/>

© 2011 - Il presente materiale non può essere utilizzato per qualsiasi fine (tra cui: riproduzione, modifica, copia, stampa, distribuzione, anche se solamente in parte) commerciale e non, senza l'autorizzazione e la citazione dell'autore.