

INTRODUZIONE

AD OMNET++

Omnet++

OMNET++ è una piattaforma di simulazione :

Modulare: gerarchia di moduli

Ad eventi

Orientata agli Oggetti (in C++)

Open Source

Versione comm. OMNEST

È utile per:

*analizzare sistemi i cui modelli sono composti da entità
che fra loro comunicano scambiandosi messaggi (utenti,
pacchetti, etc.)*

Omnet++

OMNET++ non è un simulatore
ma una piattaforma *di simulazione*,
composta da:

- λ Un motore ad eventi discreti (*kernel*),
normalmente non modificato dallo sviluppatore
- λ Una *libreria di classi* C++,
da cui normalmente lo sviluppatore *estende* le proprie

Download da: www.omnetpp.org

Omnet++

Pro e Contro

Pro:

Relativamente facile da utilizzare

Ben strutturato

Modulare

General purpose

Contro:

non molti modelli disponibili (peccato di gioventù)

Omnet++

Applicazioni principali:

Modeling of network protocols

Modeling of queueing networks

Modeling of multiprocessor systems

Performance evaluation of software systems

Frameworks



Omnet++

Componenti principali:

Kernel Libreria


Descrizione della topologia (NED)

Tkenv/Cmdenv (interfaccia utente)

Animazione e analisi dell'output

Working With OMNeT++

1. An OMNeT++ model is build from components (modules) which communicate by exchanging messages. Modules can be nested, that is, several modules can be grouped together to form a compound module. When creating the model, you need to map your system into a hierarchy of communicating modules.



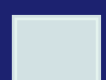
2. Define the model structure in the NED language. You can edit NED in a text editor or in the graphical editor of the Eclipse-based OMNeT++ Simulation IDE.



3. The active components of the model (simple modules) have to be programmed in C++, using the simulation kernel and class library.



4. Provide a suitable omnetpp.ini to hold OMNeT++ configuration and parameters to your model. A config file can describe several simulation runs with different parameters.



5. Build the simulation program and run it. You'll link the code with the OMNeT++ simulation kernel and one of the user interfaces OMNeT++ provides. There are command line (batch) and interactive, graphical user interfaces.

Struttura di un modello

Un *modello* viene descritto per mezzo di file .ned (NEtwork Description Language) in termini di:

- λ *moduli* (modules)
- λ *accessi* (gates)
- λ *connessioni* (connections)
- λ *messaggi* (messages)

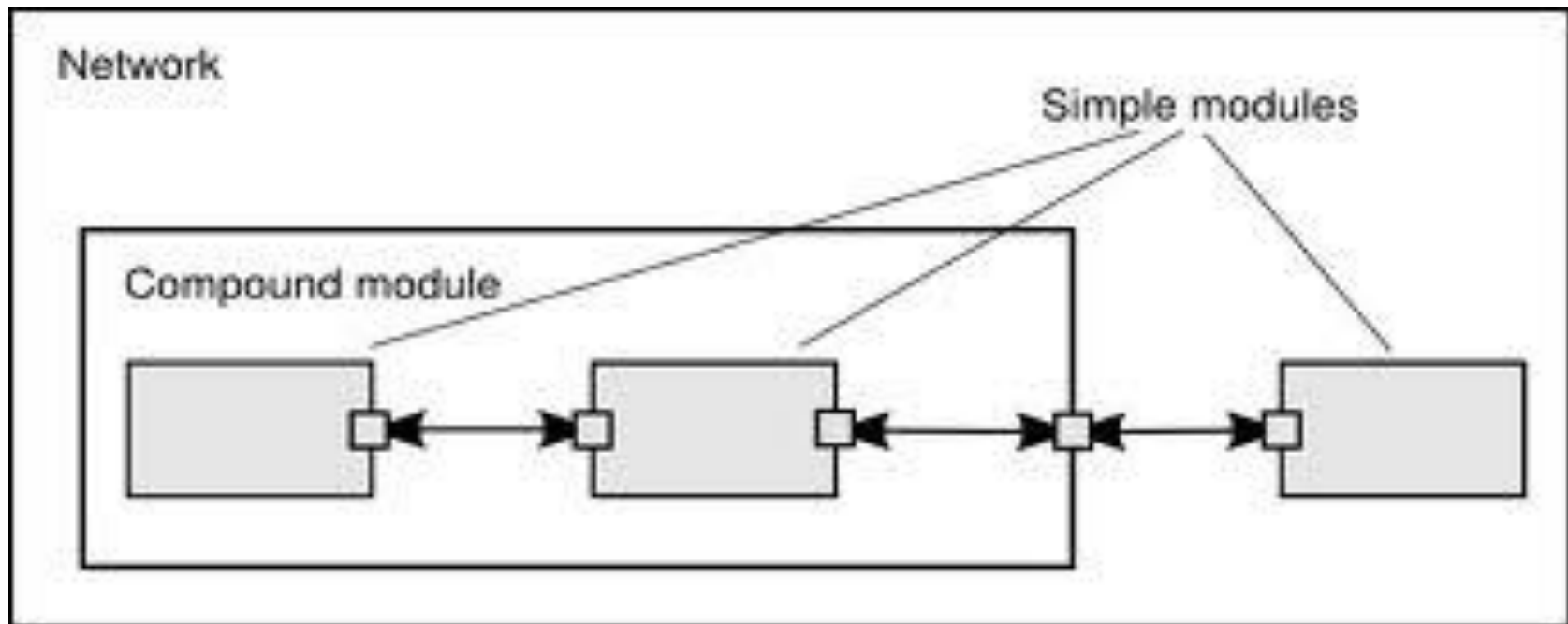
Le *simulazioni* vengono descritte per mezzo di file .ini nelle quali vengono specificati i *parametri* di moduli e connessioni e le caratteristiche della simulazione (durata della simulazione, ...)

SVANTAGGIO: questo meccanismo è rigido e *spesso* è necessario adottare metodi al di fuori di OMNET++ per ottenere quello che si vuole

Struttura di un modello

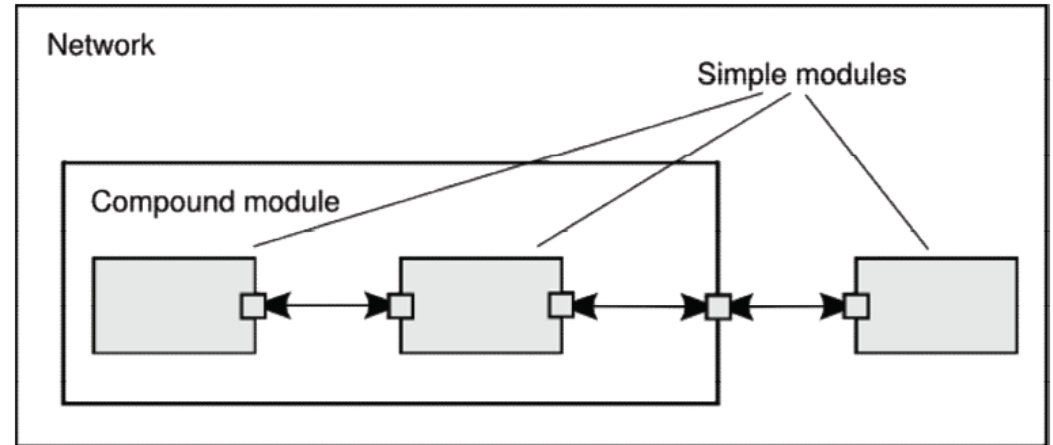
Modello: uno o più moduli connessi (network)

moduli comunicano mediante messaggi



MODULI

I **moduli** (i rettangoli) possono essere *semplici* o *composti*; La rete viene considerata un modulo composto.



Le **connessioni** (le frecce) consentono lo scambio dei messaggi tra i moduli (possono essere anche monodirezionali)

Gli **accessi** (i quadratini) sono i punti di ancoraggio delle connessioni e possono essere *in*, *out* o *inout*

MESSAGGI

I messaggi sono composti da attributi e da una qualunque struttura di dati realizzabile in C++.

I messaggi sono fondamentali in OMNET++ perché realizzano gli eventi.

I messaggi possono essere inviati dai moduli, o attraverso accessi e connessioni, oppure anche direttamente, passando per il kernel.

Con i messaggi possono essere modellati sia elementi concreti nelle comunicazioni (trame, pacchetti, bit), sia elementi logici (ad esempio la fine dell'attività di un modulo).

Omnet++

OMNET++ è relativamente giovane e ci sono delle *ambiguità* di denominazione:

- λ il prototipo di modulo viene chiamato – ma non sempre – *tipo di modulo* (module type)
- λ l'istanza di un tipo di modulo viene chiamata *modulo* (module)
- λ il prototipo di connessione viene chiamato – quasi sempre – *connessione* (connection) o *collegamento* (link)
- λ l'istanza di una connessione viene chiamata – quasi sempre – *canale* (channel)
- λ certe volte gli *accessi* (gates) vengono chiamati *porte* (ports), raramente *interfacce* (interfaces)
- λ la *libreria di classi* (class library), in determinati contesti, viene chiamata *libreria di componenti* (component library) ed in tal caso spesso i moduli, cioè i tipi di modulo, vengono chiamati *componenti* (components)

INFRASTRUTTURA

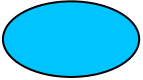

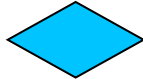

Tralasciando il kernel, l'infrastruttura di simulazione è costruita intorno alla libreria di classi:

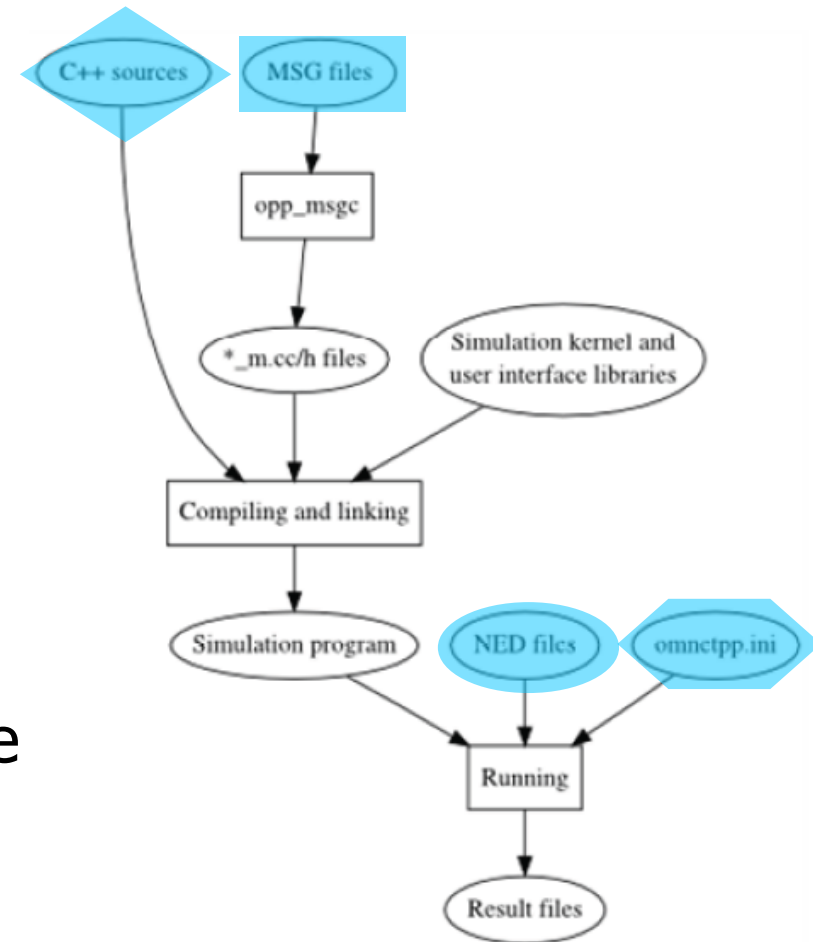
- λ moduli, accessi, connessioni
- λ parametri
- λ messaggi
- λ classi contenitore (code, vettori, ...)
- λ classi di raccolta di dati
- λ classi di stima delle distribuzioni e per la statistica
- λ classi per la rivelazione dei transitori e per la rivelazione dell'accuratezza dei risultati

Fa parte dell'infrastruttura anche il programma `opp_msgc` che permette di tradurre il formato dei messaggi in linguaggio C++

REALIZZAZIONE

Per realizzare una simulazione lo sviluppatore deve aver scritto:

- λ i file `.ned` di descrizione del modello 
- λ i file `.msg` di descrizione dei messaggi 
- λ i file `.h` e `.cc` di implementazione dei moduli semplici 
- λ i file `.ini` di descrizione della simulazione 



Eseguire una simulazione

Creare un file omnetpp.ini

configurare la simulazione

eseguire le simulazioni

Analisi dei dati

memorizzare i dati ottenuti durante la simulazione:

lunghezza delle code

Numero di pacchetti perduti

.....

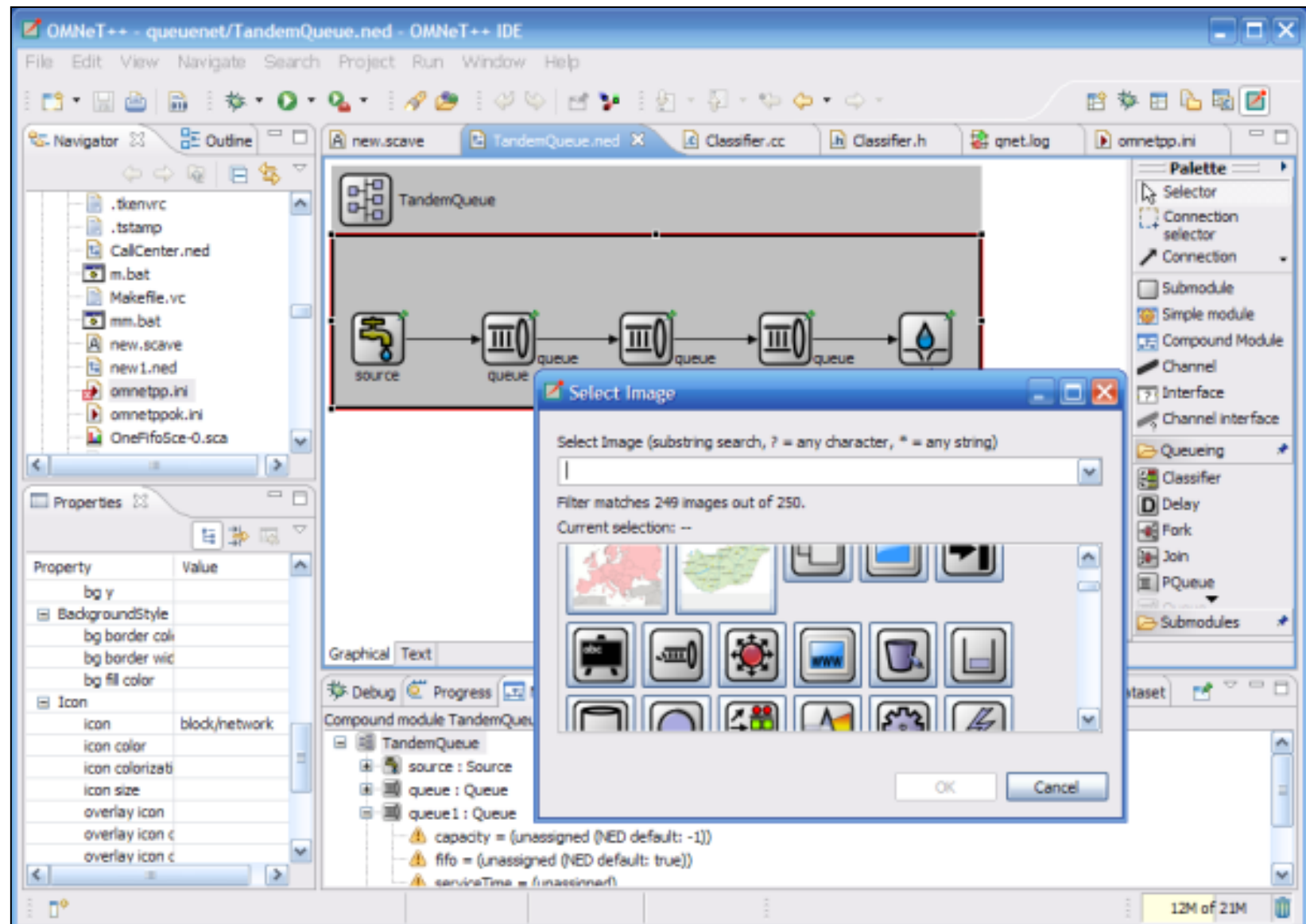
STRUMENTI

I principali strumenti a disposizione dello sviluppatore sono:

- λ l'Ambiente di Sviluppo Integrato (IDE)
basato su **Eclipse**
- λ gli Ambienti di Interfaccia Utente
uno a riga di comando,
l'altro grafico
- λ i Framework
librerie di componenti realizzate dalla comunità
OMNET++

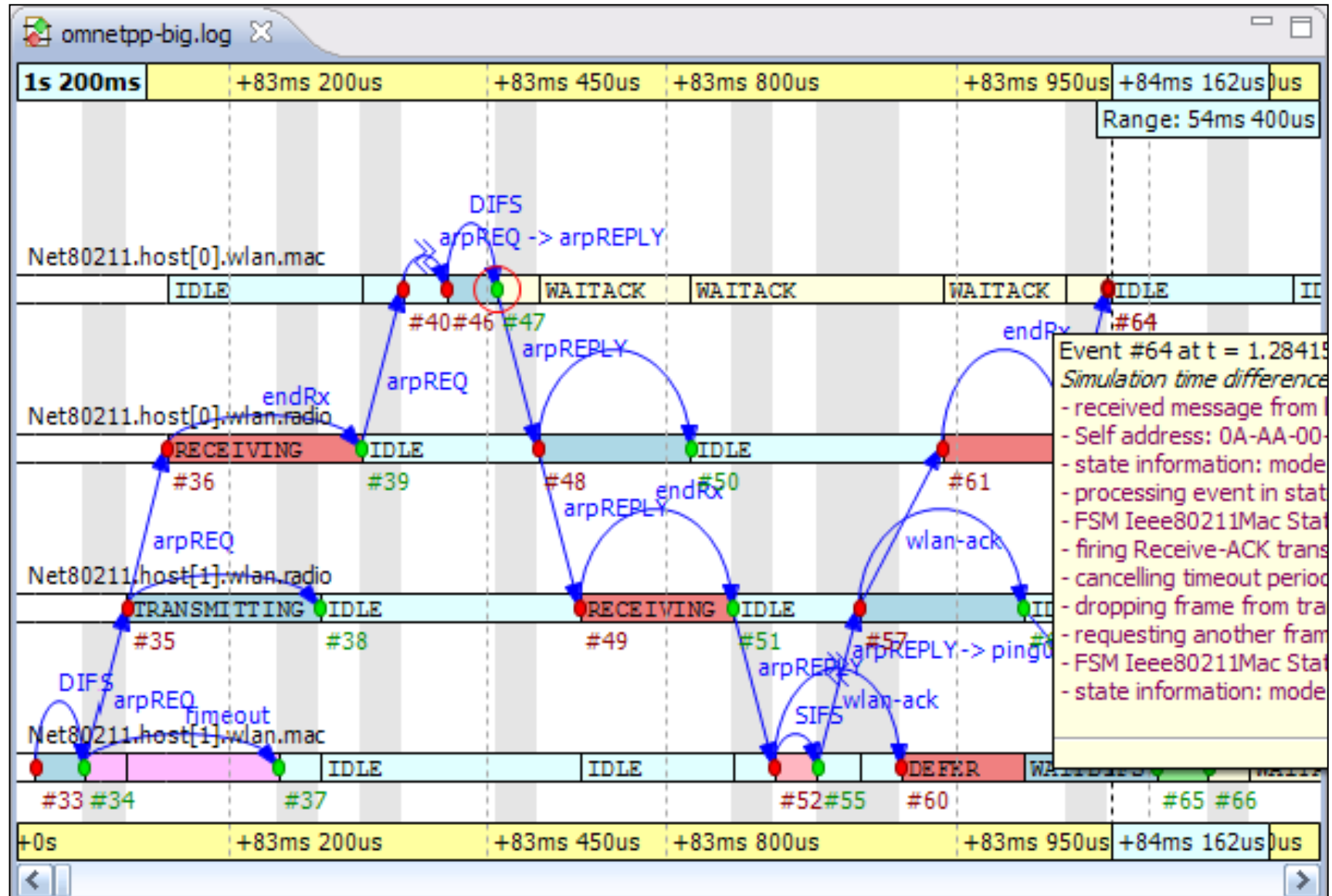
Esempio (1/3)

Utilizzo dell'editor NED in ambiente IDE



Esempio (2/3)

Diagramma di Sequenza



Esempio (3/3)

Ambiente di Interfaccia Grafica di Utente

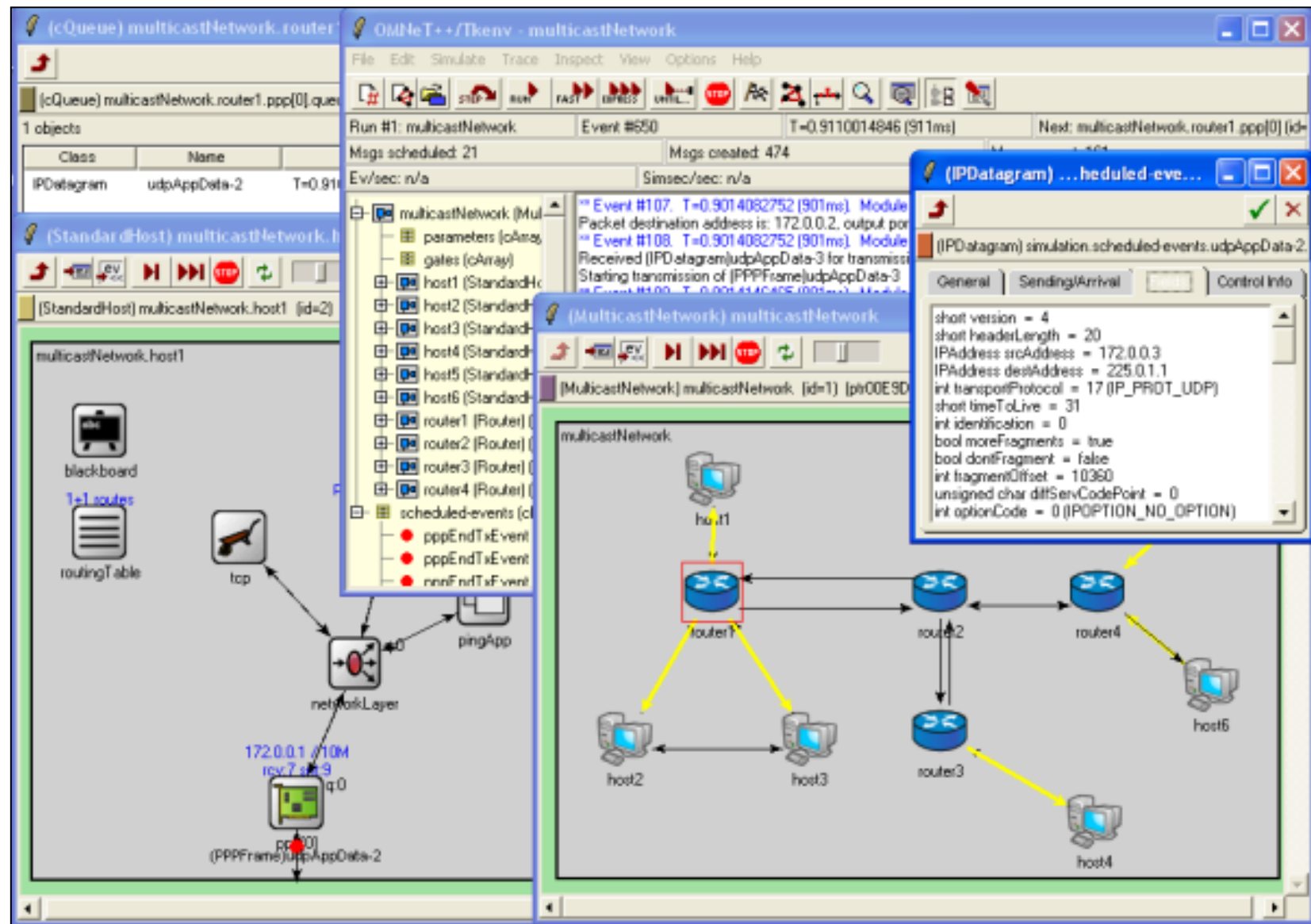


Chart generations

