

Sistemi IV

(Corso di Informatica, 5 anno)

Linguaggi, Strumenti e modelli di Simulazione

Luciano Bononi

bononi@cs.unibo.it

<http://www.cs.unibo.it/~bononi>

Ricevimento: Lun-Ven 9-18 :-)

presso ufficio dottorandi, Mura Anteo Zamboni 7, Bologna

OML: Open Modline 1.3

- Modline 1.3: definisce un ambiente di simulazione (OML) oltre che un tool-set per la simulazione di sistemi a eventi discreti
- Tool-set
 - problem analysis
 - model development
 - simulation
 - result display
 - report generation

OML: Open Modline

- Componenti e struttura dell'ambiente
 - Qnet: package grafico per editing, solving, animazione di reti di code
 - Qnap2: linguaggio per costruire il modello (librerie e tipi di oggetti)
 - Ambiente: tool generici di OML
 - Experimenter: gestione esecuzioni di modelli e specifica di processi sperimentali
 - Analyzer: gestione risultati e grafici ottenuti da sim.
 - Reporter: gestione presentazione risultati
 - Workbench grafico: interfaccia utente per l'ambiente, manipolazione oggetti (no command line)

OML: Open Modline

- Componenti e struttura dell'ambiente
 - Ambiente (continua): Tool per uso avanzato
 - Library browser: gestione librerie di oggetti
 - Type editor
 - Model configurator

OML: Open Modline

- Documentazione
 - Modline manuals
 - OML user guide
 - 1 OML Installation
 - 2 OML primer
 - 3 Experimenter User Guide
 - 4 Analyzer User Guide
 - 5 Reporter User Guide
 - 6 OML reference manual
 - Modline User Guide
 - 7 Qnet User Guide
 - 8 Qnap2/Qmodule User Guide

OML: Open Modline

- Documentazione
 - Modline manuals
 - Advanced Modline User Guide (tool avanzati: Qobjects)
 - 9 Qobjects Intro
 - 10 Library Browser
 - 11 Type editor
 - 12 Model Configurator
 - Modline Tutorials
 - 13 Tutorial Introduction
 - 14 Modline tutorial
 - 15 Advanced Modline tutorial
 - 16 Model Configurator tutorial

OML: Open Modline

- Documentazione
 - Qnap2 manuals
 - Qnap2 9.3 User guide
 - caratteristiche di Qnap2
 - meccanismi di Qnap2 per definire, costruire e analizzare modelli a reti di code
 - Qnap2 9.3 Reference Manual (Tome I e Tome II)
 - descrizione dettagliata di TUTTI i comandi del linguaggio
 - Gnuplot manuals
 - Makefile Manuals

OML: Open Modline

- Open Modline Primer
 - New User Environment setup
 - `cd /scsusr/software/simulog/modline/`
 - `make user`
 - crea `.modlinerc.csh` nella propria home
 - » settare GNUmake e non `/bin/make/`
 - » preferenze nei tool di editing, grafici, ecc.
 - attenzione a salvare prima una copia di `.cshrc` (ecc.)
 - tornare alla home (`cd`) ed eseguire “`source .cshrc`”
 - `make dir NAME=<nome della dir>`
 - `tkwb` oppure `xdtm` (Xwindows desktop manager)

OML: Open Modline

- L'ambiente grafico permette tutte le operazioni fattibili da command line
 - Selezionare un oggetto: left click
 - attivare o entrare in un oggetto: double left click
 - da quel momento ogni comando o operazione contestuale è riferita all'oggetto in cui mi trovo (es. make edit, make run, make display...)
 - selezione di comandi, opzioni o valori per l'oggetto: right click sull'oggetto, shift e rilascio sull'opzione.

Qnet: creazione nuovo modello

- Nuovo modello (source-sink)
 - creiamo un nuovo modello con Qnet
 - `make dir NAME=<nome>`
 - `cd <nome>.dir`
 - la directory viene considerata da Modline come un oggetto di tipo “dir”: accedervi significa entrare nel contesto delle possibili operazioni per quell’oggetto.
 - Es. creiamo un modello qnet in questa dir
 - `make qnet NAME=<name>` (crea `<name>.qnet`)
 - `cd <name>.qnet` (entra nel contesto dell’oggetto creato)

Qnet: operazioni sul modello

- Operazioni possibili su modelli (qnet objects)
 - click destro su <nome>.qnet
 - edit: crea o modifica un modello, e consente di generare il codice Qnap2 relativo
 - run: consente di eseguire una singola esecuzione del modello (richiedendo eventuali valori di input)
 - test: consente diverse esecuzioni dello stesso modello (non modificato) al variare di parametri di input
 - meter: mostra lo stato di esecuzione del/dei modello/i eventualmente connessi a un determinato PLAN
 - display: mostra i risultati di un run (tabella o grafico)
 - anim: anima il grafo del modello per il run (richiesta preventiva)

Qnet: operazioni sul modello

- Edit (su oggetto .qnet)
 - selezionare ICONA oggetti (es. source, sink)
 - posizionare gli oggetti nel grafo
 - determinare gli attributi degli oggetti (nodi): click destro sull'oggetto selezionato, “set node attributes”
 - N.B. nomi di oggetti: MAX 8 char significativi!!!
 - determinare le statistiche richieste per l'oggetto: “statistical results subform”
 - selezionare “generate code & specs” del modello per la traduzione in Qnap2.
 - Save & exit

Qnet: operazioni sul modello

- Run (su oggetto .qnet)
 - vedremo Experimenter per gestire i PLAN
 - viene eseguito solo se “code & specs” del modello sono “nuovi” rispetto all’esecuzione precedente
 - il codice Qnap2 si trova nel file compiled.qnp
 - eseguendo run sul modello, si apre un log file dell’esecuzione (qnap2.log) contenente eventuali errori.
 - L’esecuzione genera
 - output.result (formato DI)
 - output.tl (usato per display)
 - animation_trace.tl (solo se richiesto)

Qnet: operazioni sul modello

- **Test**
 - viene eseguito anche se “code & specs” non sono nuovi
 - utile per test e confronto dei risultati con dati diversi
- **Meter**
 - definisce lo stato di avanzamento del progetto (plan) di simulazione
 - E’ testuale e grafico (model meter)
- **Display**
 - visualizza i dati previsti come risultato di simulazione
 - apre output.result e visualizza (o rende graficamente) i dati di simulazione
 - apertura file: read only

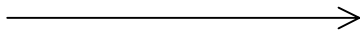
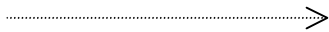
Qnet: operazioni sul modello

- Anim
 - mostra l'animazione dell'esecuzione della simulazione
 - vale solo se il metodo di soluzione è Simulazione
 - l'animation trace deve essere richiesto preventivamente
 - il run di simulazione deve essere completato
- generate_report
 - genera documentazione del modello (HTML, Latex, RTF)

Qnet: operazioni sul modello

- Modelli = grafi
 - nodi e link tra nodi (+ commenti)
 - ognuno è istanza di una classe (*non Qnap2) che determina
 - tipo degli attributi
 - connettività (con chi, in che modo, ecc.)
 - sottografi definiti
 - visualizzazione
 - Attributi: valori associati a un oggetto
 - Connettività: (nodo-nodo, link-nodo, link-link)
controllata attraverso PORTs (specificano quali tipi di oggetti possono essere attaccati a un altro oggetto)

Qnet: operazioni sul modello

- Modelli = grafi
 - Appearance: icone degli oggetti, testo, colore, ecc.
 - tipo dell'oggetto (classe)
 - valori degli attributi (es. Fifo, Lifo)
 - valori degli attributi del grafo in cui l'oggetto è inserito
 - Links: sono rappresentati come linee
 - link di flusso 
 - link di riferimento 
 - Piani: rappresentano visioni contestuali di modelli
 - hanno un nome e ne esistono di predefiniti
 - Transizioni (nodi e links), synchronization (resource/split(match),
 - Graphical annotation (sink, references), animation (meters)...

Qnet: operazioni sul modello

- Tipi di dati per gli attributi
 - Tipi semplici
 - integer, integer range
 - real, real range
 - string (una riga di testo)
 - text (tante righe)
 - choice item (tipi enumerati Pascal like)
 - tipi complessi
 - record: tanti field di tipo fissato
 - list: lista di elementi dello stesso tipo
 - Union indicizzata: scelta di una tra una lista di strings
 - FREE values: valori non ancora definiti:
 - label, info text, (e tipo del valore che può assumere)

Qnet: operazioni sul modello

- Gerarchia grafica
 - sottografi (oggetto submodel)
 - edit submodel:
 - internal submodel
 - external submodel (rispetto al modello generale viene salvato fuori)
 - delete submodel (cancella il contenuto ma non rimuove l'oggetto)
 - non posso definire “Set submodel attributes”
 - non posso generare “code & specs” per submodels
 - Port esportati: sono nodi “grafici” per collegamento
 - Set binding names: dare nomi ai port per risolvere ambiguità
 - N.B. creazione modelli bottom-up!

Qnet: operazioni sul modello

- Interfaccia grafica Qnet
 - settare le variabili di ambiente preferite (se volete)
 - Mouse. (...annoyance)
 - Menubar. (...annoyance)
 - dump whole model (salva il grafo in epsf file)
 - graph (model) object menu (click dx su backgnd)
 - set model attributes, commenti, clipboard, check
 - generate code & specs !!!
 - canvas object (nodi, link, annotation) menu (click dx su oggetto): save, save & exit
 - contiene l'elenco delle operazioni per l'oggetto
 - Set node attributes, Make link, attach subgraph, flip...

Qnet: operazioni sul modello

- Create node
 - Source
 - sorgente infinita di utenti
 - arrival rate = service demand
 - possibilità di avere definizioni di arrivi algoritmici
 - name: station identifier
 - server features: CST, EXP, HEXP, ERLANG, UNIFORM, RINT, COX, USER
 - statistical results: ...
 - link in uscita:
 - new flow, continued (verso divert nodes), reference (pointer)
 - link in ingresso:
 - origin (da match nodes), check (per animazioni)

Qnet: operazioni sul modello

- Create node
 - Sink
 - è un nodo grafico
 - implica la distruzione del customer
 - link in ingresso
 - flow (da server), new_flow (da sorgenti), created (da split), diversion (da server a capacità limitata), continued (da convert)
 - nessun link in uscita

Qnet: operazioni sul modello

- Create node
 - Divert
 - nodo grafico
 - aggrega flussi multipli in uscita da un singolo nodo
 - nodo `<--continued-->` Divert `<<<<` tanti link verso tanti nodi
 - link in ingresso
 - continued (da nodo singolo)
 - reference (da transizioni algoritmiche)
 - link in uscita
 - flow (verso server), created (verso split), continued (verso convert), reference (per transizioni algoritmiche)

Qnet: operazioni sul modello

- Create node
 - Convert
 - nodo grafico
 - aggrega flussi multipli in ingresso a un singolo nodo
 - tanti link da tanti nodi >>> Convert <--continued--> nodo
 - link in ingresso
 - flow (da server), created (da split), continued (da convert), reference (da transizioni algoritmiche)
 - link in uscita
 - continued (per nodo singolo)
 - reference (per transizioni algoritmiche)

Qnet: operazioni sul modello

- Create node
 - Server station
 - esegue servizi definiti in “Service” e “Service x class”
 - i customers che completano il servizio seguono il routing probabilistico dei link in uscita (flow)
 - Transit o Move sono procedure algoritmiche che possono forzare transizioni prima del completamento del servizio

Qnet: operazioni sul modello

- Create node
 - Server station
 - name: identificatore
 - Server features
 - » Service x class
 - » Service (default): CST, EXP, HEXP...
 - » Multiplicity: serve a dichiarare una stazione multiserver
 - » population dependent service rate
 - Queue features
 - » customer init: numero iniziale di customers nel server
 - » priorities
 - » capacity (limite del buffer): se Finita: total limit, default reject policy (skip=passa al nodo successivo, divert link=passa al nodo specificato dall'unico divert link, User code=codice algoritmico in Qnap2)

Qnet: operazioni sul modello

- Create node
 - Server station
 - Sched: Fifo, Lifo, Quantum, PS
 - statistical results: da richiedere (vedi tabella)
 - link in ingresso:
 - flow (da server), new_flow(da sorgenti), created (da split), continued (da divert o convert), reference (algs), origin (match)
 - link in uscita
 - flow (verso server), new_flow (solo verso split!), diversion (limite di capacità raggiunto), continued (divert o convert), reference (algs)
 - monoserver, multiple server, infinite server (delay node)

Qnet: operazioni sul modello

- Create node
 - Split node
 - un customer viene suddiviso in un set di customer
 - tipi:
 - split = tiene traccia dell'origine (customer che ha originato il set)
 - fission = non tiene traccia dell'origine (set di customers qls.)
 - N.B. il customer che entra nello split viene **DISTRUTTO** e quindi attenzione a eventuali join che lo attendono
 - in seguito un nodo match può attendere tutti i customer splittati e al loro arrivo (completo) genera un solo nuovo customer che prosegue nel grafo.

Qnet: operazioni sul modello

- Create node
 - Match node:
 - match o fusione di customers ottenuti da uno split
 - viene rigenerato un unico customer
 - tipi:
 - match: ricombina i customers di uno split precedente
 - ordinary fusion: ricombina un set proprio di customers qls.
 - Gli incoming customers sono distrutti! (occhio ai join)
 - N.B. Split va usato con match, ma fission e fusion sono generali e possono essere usati in modo indipendente

Qnet: operazioni sul modello

- Create node
 - Resource node
 - sono oggetti passivi (es token presi e rilasciati)
 - servono per modellare risorse condivise
 - sono semafori! Azioni P e V.
 - un reference link rappresenta un pointer da una stazione alle sue risorse
 - multiplicity: il numero di token iniziale nel pool
 - priorità e scheduling: fifo, lifo, prior, preempt...
 - N.B. le risorse, se usate, implicano che il modello può essere risolto solo con simulazione (extended QN)
 - solo link reference possono entrare/uscire in risorse

Qnet: operazioni sul modello

- Create node
 - Resource node
 - attributi
 - Nome
 - server features (multiplicity= num risorse all'inizio della sim.)
 - queue features (priorities quando lo scheduling è PRIOR) e (SCHEDuling discipline)
 - Statistical results (utilizzo risorse, ecc.)
 - Power: default ON = mostra il numero di risorse disponibili in animazione.

Qnet: operazioni sul modello

- Create node
 - User statistics (nodo stat_var)
 - variabili definite dall'utente, calcolate, e stampate
 - Solo un carattere di _ è consentito nel nome!!!

Qnet: operazioni sul modello

- Link types e loro attributi
 - Flow:
 - esprime flusso probabilistico e classe destinazione
 - solo per rappresentare flussi statici
 - flow x class
 - flow: From (classe di origine) To (classe destinazione) Prob. (probabilità del transito)
 - New_flow
 - esprime flusso probabilistico (a partire da una sorgente, o da server a split) e classe destinazione
 - solo per rappresentare flussi statici
 - monaclass: probability subfield
 - multiclass: Origin, Destination Class, prob. Per ogni class

Qnet: operazioni sul modello

- Link types e loro attributi
 - created
 - path di un nuovo customer creato in uno split
 - created customer: specifica una classe e un numero di customer
 - diversion
 - path di customer respinti da un server (limite capacità)
 - continued
 - per andare in nodi Divert o uscire da nodi Convert
 - origin
 - specifica da quali code si ottengono i customers per match o fusion, oppure è usato da match a split nodes
 - Origin classes: lista di tutte le classi di origine per i customer

Qnet: operazioni sul modello

- Link types e loro attributi
 - check
 - connette un oggetto display generico a un oggetto del modello
 - attributi:
 - integer display:
 - » current nb of customers
 - » Total nb customers entered, exited
 - real display:
 - » Mean busypct, service time, response time, custnb, throughput, blocked time
 - reference
 - è solo link grafico informativo (non genera nulla in Qnap)