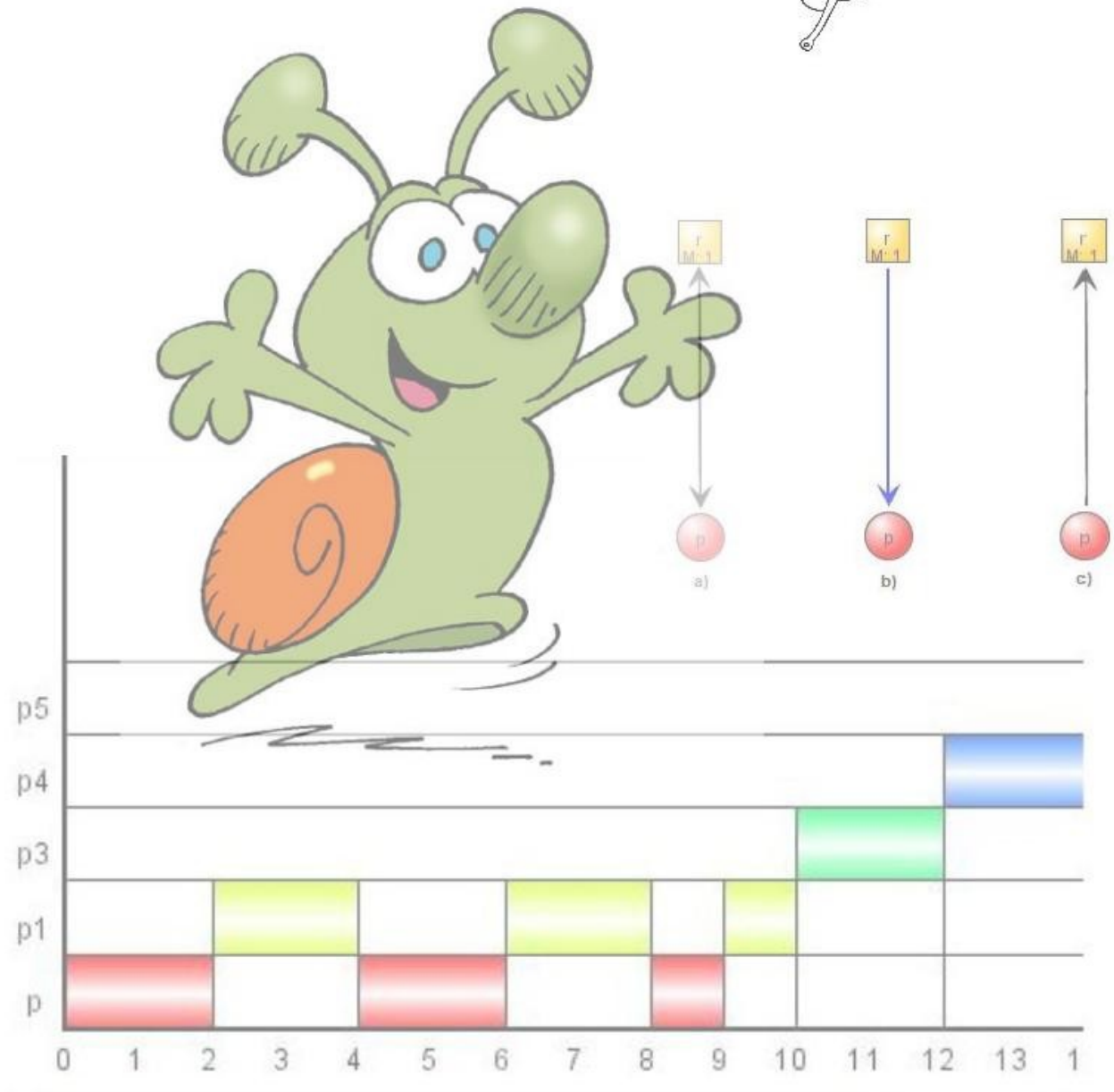
Snail Systems

Manuale d'uso

S.G.P.E.M v2.0



Snail Systems srl
Sede Legale: via Belzoni, 12 – 35131Padova
Indirizzo web: **www.snailsystems.altervista.org** E-mail: **snailsystems@yahoogroups.com**

Indice Contenuti

1. Introduzione 3
    1.1 Definizione dell'utente del prodotto 3
    1.2 Come leggere il Manuale 3
    1.3 Documenti utili 3
    1.4 Come riportare problemi e malfunzionamenti 3
2. Descrizione Generale 4
    2.1 Caratteristiche del prodotto 4
        2.1.1 Linguaggio XML 4
        2.1.2 Modello XML 5
            2.1.2.1 Struttura del modello 5
    2.2 Requisiti di sistema 5
        2.2.1 Requisiti software 6
        2.2.2 Requisiti hardware minimi 6
        2.2.3 Requisiti hardware consigliati 6
3. Istruzioni d'uso 6
    3.1 Avvio del prodotto 6
    3.2 Scelta iniziale 6
    3.3 Nuova Simulazione 7
        3.3.1 Modalita' Wizard 7
            3.3.1.1 Scelta Politica 8
            3.3.1.2 Inserimento Processi 9
            3.3.1.3 Inserimento Risorse 9
            3.3.1.4 Inserimento Richieste 10
        3.3.2 Modalita' TextMode 10
            3.3.2.1 Funzionalita' Editor 11
    3.4 Carica Simulazione 12
        3.4.1 Modalita' Grafica (VisualData) 12
        3.4.2 Modalita' Testuale (TextMode) 12
    3.5 Esecuzione 12
    3.6 Barra dei comandi 12
        3.6.1 Barra modalita' Wizard 14
        3.6.2 Barra modalita' TextMode 14
        3.6.3 Barra di esecuzione 15
    3.7 Gestione simulazione 15
        3.7.1 Avanzamento processi 16
        3.7.2 Assegnazione risorse 16
        3.7.3 Coda dei processi 17
        3.7.4 Simulazione testuale 17
        3.7.5 Statistiche 17
    3.8 Salvare la Simulazione 18
        3.8.1 Salvare l'ambiente di simulazione 18
        3.8.2 Salvare i risultati dell'esecuzione 18
4. Teoria dei processi 18
    4.1 Comunicazione e sincronizzazione tra processi 18
        4.1.1 Monitor 18
        4.1.2 Barriere 19
    4.2 Deadlock(stallo) 19
        4.2.1 Prevenzione 19
        4.2.2 Riconoscimento e recupero 19
        4.2.3 Indifferenza 19
5. Politiche di ordinamento 20
    5.1 Programmazione concorrente 20
    5.2 Classificazione dei sistemi 20
    5.3 Politiche di ordinamento 20
        5.3.1 Per sistemi a lotti 20
        5.3.2 Per sistemi interattivi 21
    5.4 Inversione di priorita' 21
        5.4.1 Condivisione di risorse 21
        5.4.2 Il problema 22
        5.4.3 Soluzioni 22
6. Aggiunta nuove politiche di schedulazione 22
    6.1 Creazione dela classe 23
    6.2 L'interfaccia Policy 23
    6.3 Compilazione della classe 26

1. INTRODUZIONE

Questo manuale descrive l'utilizzo del software S.G.P.E.M. V2 per la gestione processi in un elaboratore multiprogrammato.

1.1 Definizione dell'utente del prodotto

Il prodotto S.G.P.E.M. v2 e' orientato all'utilizzo didattico sia da parte degli studenti di informatica (laurea triennale) nel corso di Sistemi Operativi sia dai docenti per effettuare simulazioni durante le lezioni e per rendere piu' chiari i concetti relativi alla gestione processi processi e risorse in un elaboratore multiprogrammato.

1.2 Come leggere il manuale

Il presente manuale descrive punto per punto le operazioni principali del software al momento dell'avvio. In ordine vengono descritte le fasi di inserimento, di esecuzione e visualizzazione dati. La descrizione viene semplificata dalla visione di immagini rappresentanti l'interfaccia principale utili per la veloce comprensione sulla selezione comandi e sulle loro funzionalita'. Per una ricerca rapida dell'argomento voluto fa riferimento l'indice precedente.

1.3 Documenti utili

Per una maggiore comprensione si raccomanda la visione dei seguenti documenti:

- Glossario.pdf v1.2;

1.4 Come riportare problemi e malfunzionamenti

Compilare la seguente tabella per notificare eventuali errori o malfunzionamenti del software S.G.P.E.M. v2

Versione prodotto	
Hardware in uso	
Sistema operativo	
Fase operativa	• Nuova Simulazione • Caricamento Simulazione • Esecuzione Simulazione • Visualizzazione dati
Modalita' di utilizzo	• Wizard • Editor Testuale • Grafici • Statistiche
Tipo di Errore	

Versione prodotto	
Descrizione del problema	
Note	

2. DESCRIZIONE GENERALE

Il software S.G.P.E.M. v2 si propone di simulare la schedulazione di processi software. L'uso si divide in due fasi principali che consistono nella creazione dell'ambiente di simulazione tramite modalita' grafica (Wizard) o modalita' testuale (TextMode) e dall'esecuzione e consultazione di questo tramite grafici e statistiche.

L'ambiente di simulazione consiste in un file XML che viene interpretato e verificato da un apposito parser.

2.1 Caratteristiche del prodotto

- Creazione ambiente di simulazione in modalita' grafica;
- Creazione ambiente di simulazione in modalita' testuale;
- Salvataggio ambiente di simulazione su file testuale XML;
- Caricamento ambiente di simulazione in modalita' grafica;
- Caricamento ambiente di simulazione in modalita' testuale;
- Transizione da modalita' grafica a modalita' testuale;
- Transazione da modalita' testuale a modalita' grafica;
- Esecuzione ambiente di simulazione;
- Visualizzazione dati in modalita' "automatica", "step-by-step" ed a "selezione di istante";
- Visualizzazione grafo temporale;
- Visualizzazione statistiche su grafici;
- Visualizzazione statistiche su testo;
- Help contestuale richiamabile;
- Importazione politiche di schedulazione.

2.1.1 Linguaggio XML

L'Extensible Markup Language (XML) è un metalinguaggio che permette di creare dei linguaggi personalizzati di markup; nasce dall'esigenza di portare nel World Wide Web lo Standard Generalized Markup Language (SGML), lo standard internazionale per la descrizione della struttura e del contenuto di documenti elettronici di qualsiasi tipo; ne contiene quindi tutta la potenza, ma non tutte le complesse funzioni raramente utilizzate. Si caratterizza per la semplicita' con cui è possibile scrivere documenti, condividerli e trasmetterli nel Web.

In S.G.P.E.M. v2 e' stato definito un metalinguaggio XML per la creazione ed il salvataggio di un ambiente di simulazione al fine di semplificarne la progettazione ed al contempo stesso fornire all'utente un utile strumento per un completo utilizzo del software.

2.1.2 Modello XML

Segue la struttura del linguaggio XML per l'inserimento dei dati relativi all'ambiente di simulazione. Lo stesso modello puo' essere richiamato dall'utente in modalita' TextMode [cfr. §3.3.2].

2.1.2.1 Modello XML adottato:

```
<?xml version="1.0"?>
<!DOCTYPE simulation SYSTEM "Sgpem2Input.dtd">
<simulation title="NONAME">

    <policy name=""  />

    <process      name=""      arrival=""      execution=""    priority="" />

    <resource     name=""      multiplicity="" arrival="" />

    <request      process=""   resource=""     arrival=""      execution="" />

</simulation>
```

Il modello sopra comprende un'intestazione iniziale con la versione del linguaggio ed il file con la descrizione della grammatica. Il corpo della simulazione e' delineato dal tag <simulation>; l'attributo title permette di inserire il titolo della simulazione. Gli altri tags con i relativi attributi sono (gli attributi segnati con "*" sono obbligatori):

- <policy>: serve per inserire la politica di schedulazione e contiene gli attributi *name**, ove inserire il nome della politica, e *timeslice* per l'intero rappresentante il timeslice (obbligatorio se richiesto dalla politica prescelta);
- <process>: identifica un processo attraverso gli attributi *name** per il nome del processo, *arrival** per il tempo di arrivo, *execution** per il tempo di esecuzione, *priority* per la priorita' (obbligatoria se richiesta dalla politica prescelta);
- <resource>: identifica una risorsa attraverso gli attributi *name** per il nome della risorsa, *multiplicity* per la molteplicita' (obbligatoria se richiesta dalla politica prescelta), *arrival** per il tempo di arrivo;
- <request>: identifica una richiesta di un processo per una risorsa; e' caratterizzata dagli attributi *process** per il relativo processo, *resource** per la risorsa, *arrival** per il tempo di arrivo, *execution** per il tempo di esecuzione.

2.2 Requisiti di sistema

Il software S.G.P.E.M. v2 necessita dei seguenti requisiti per un corretta esecuzione.

2.2.1 Requisiti software

Sistema operativo (testato) :

- Linux: Tutte le piattaforme i565 e 100% compatibili con qualsiasi distribuzione.
- Windows:
  - Windows95;
  - Windows98;
  - WindowsME;
  - Windows2000;
  - WindowsXp home/professional;

E' necessaria la presenza nel sistema usato del pacchetto Sun Java Runtime Environment (JRE) v1.5.0 o superiore.

2.2.2 Requisiti hardware minimi

- Processore Intel/AMD (o 100% compatibile) 200Mhz;
- 64MB di memoria Ram;
- 100MB di spazio libero su disco;

2.2.3 Requisiti hardware consigliati

- Processore Intel/AMD (o 100% compatibile) 1000Mhz;
- 128MB di memoria Ram;
- 200MB di spazio su disco;

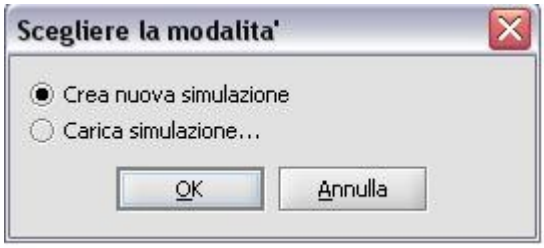
3. ISTRUZIONI D'USO

3.1 Avvio del prodotto

Per avviare il programma, basta selezionare il file sgpemv2.jar dalla directory SGPEMv2 oppure tramite prompt dei comandi del proprio Sistema operativo digitando, dalla medesima directory, il comando "java -jar sgpemv2.jar". Dopo un breve tempo di caricamento comparira' a su schermo la finestra iniziale del software.

3.2 Scelta iniziale

All'avvio del programma l'utente ha a disposizione una serie di scelte per iniziare ad utilizzare il programma; le possibili scelte sono:



- fig. 3.2 -

Le stesse scelte possono essere richiamate dal menu' "File" della finestra principale o dai singoli tasti sulla barra dei comandi [vd. fig. 3.6x].

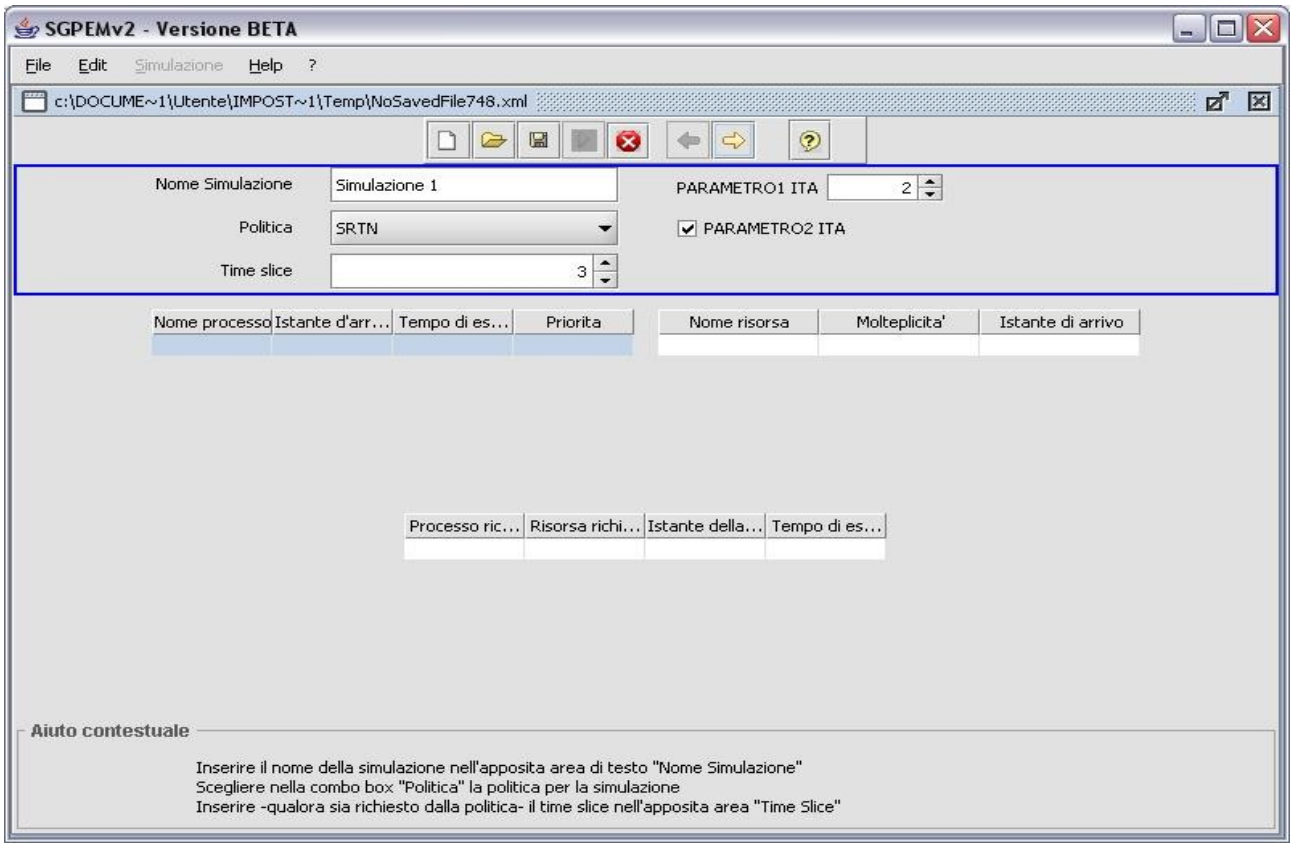
3.3 Nuova Simulazione

Scegliendo "Crea nuova Simulazione" si avvia la procedura di creazione di un nuovo ambiente di simulazione da mandare in esecuzione subito o in un secondo momento. Le modalita' di creazione di un nuovo ambiente di simulazione sono due: modalita' grafica (Wizard) e modalita' testuale (TextMode).

- fig. 3.3 -

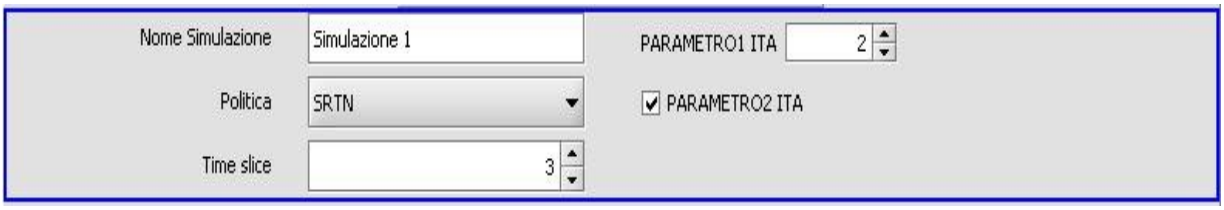
3.3.1 Modalita' Wizard

Scegliendo di creare una nuova simulazione tramite il Wizard, si apre una schermata nella quale sono presenti tutti i pannelli e i comandi per l'inserimento completo dei dati di un'intera simulazione. In fondo e' presente un'area che aiuta l'utente ad ogni passo.



- fig. 3.3.1 -

3.3.1.1 Scelta della Politica



- fig. 3.3.1.1 -

All'inizio sara' attivo solo il pannello per la definizione della politica, che comprende:

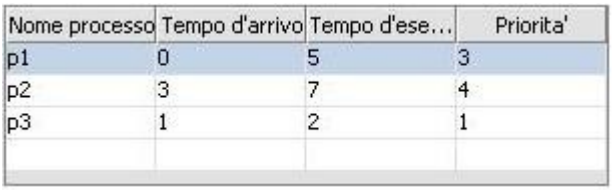
- un'area per il nome della simulazione;
- una combobox per selezionare la politica;
- checkbox e contatori per parametri aggiuntivi relativi alla singola politica di schedulazione. (come ad esempio il time_slice) .

Cliccando sul tasto "Avanti" (freccia destra sulla barra dei comandi) e' possibile passare alla fase successiva di inserimento dati, ovvero quella di inserimento di processi, risorse e richieste. Tale operazione comporta la disattivazione del pannello di definizione della politica (che rimane comunque visibile) e l'attivazione del pannello per l'inserimento di processi, risorse e assegnazioni.

NOTA: non e' possibile avanzare al passo successivo se non e' stata selezionata una

politica.

3.3.1.2 Inserimento Processi



- fig. 3.3.1.2 -

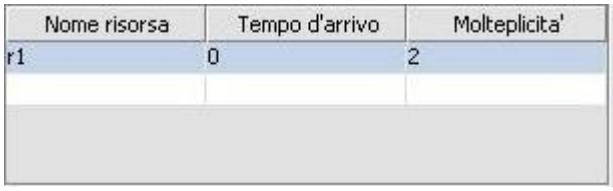
Un processo e' caratterizzato da:
- un nome che lo identifica;
- l'istante di arrivo nella coda dei processi pronti;
- il tempo di esecuzione;
- un'eventuale priorita' (nel caso la politica lo richieda).

La tabella di inserimento processi permette appunto di inserire queste caratteristiche. Nel caso la politica selezionata al punto precedente sia senza priorita', la colonna della priorita' e' disattivata. Quando si e' completata una riga, ovvero si ha definito completamente un processo, e' possibile definirne un altro inserendo i relativi dati nella riga vuota successiva.

E' possibile cambiare politica anche dopo aver inserito alcuni processi, cliccando sul tasto "Indietro" [cfr. § 3.3.1.1]. Il cambio della politica scelta non comporta la cancellazione dei processi inseriti. Se la nuova politica comporta di inserire dei dati aggiuntivi non precedentemente inseriti, come la priorita', tale inconsistenza verra' gestita tramite opportune segnalazioni all'utente.

NOTA: non e' possibile avanzare alla tabella successiva se non e' stato inserito almeno un processo.

3.3.1.3 Inserimento Risorse

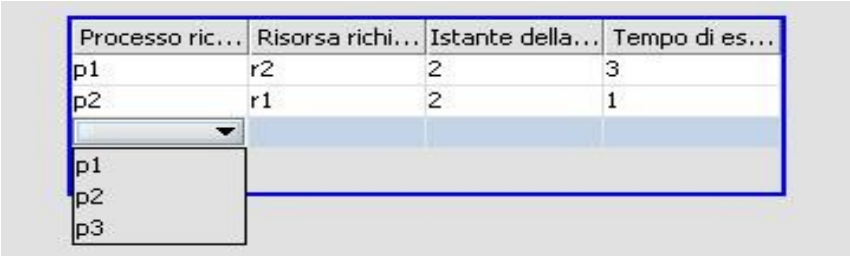


- fig. 3.3.1.3 -

Una risorsa e' caratterizzata da:
- un nome che la identifica;
- un tempo d'arrivo;
- una molteplicita'.

In questa sezione non bisogna inserire la CPU. Essa e' infatti considerata come risorsa principale e non escludibile. Questa sezione e' pensata per l'inserimento delle risorse aggiuntive alla CPU. Nel caso non si inserisca nessuna risorsa, l'applicazione crea una simulazione nella quale i processi sono schedulati per la sola esecuzione nella CPU.

3.3.1.4 Inserimento Richieste



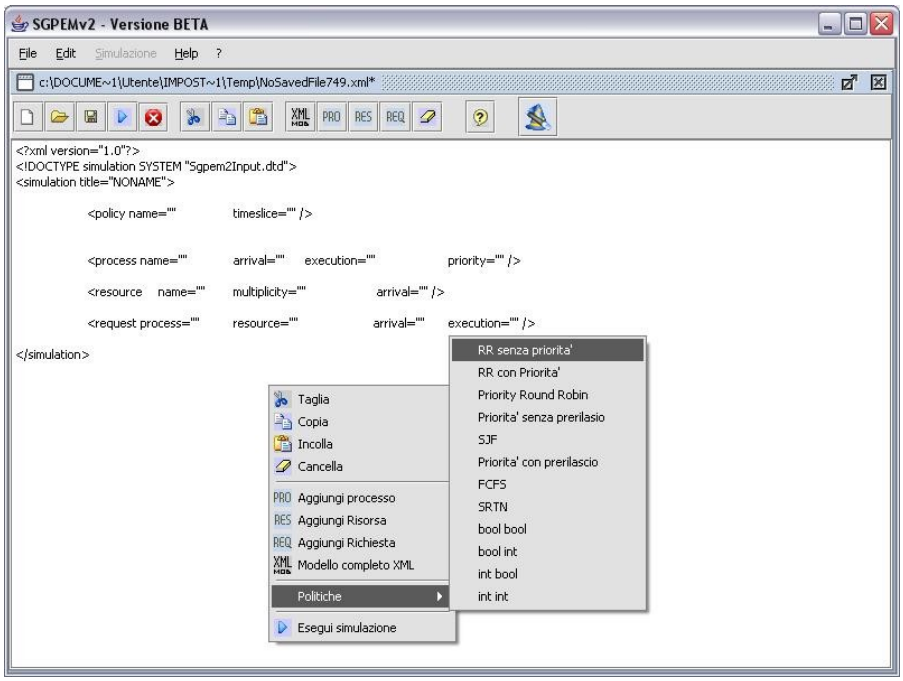
- fig. 3.3.1.4 -

La tabella di inserimento richieste processo-risorsa e' attiva solo se sono state inserite delle risorse nella relativa tabella.
Una richiesta e' caratterizzata da:

- il nome del processo che richiede la risorsa, selezionabile da una lista dei processi inseriti in precedenza;
- il nome della risorsa che viene richiesta dal processo, selezionabile da una lista delle risorse inserite in precedenza;
- l'istante in cui tale richiesta avviene (a partire dal primo istante di esecuzione del processo). (Es.: Se P1 inizia la sua esecuzione all'istante 3 ed effettua una richiesta di R1 con tempo di arrivo pari a 2 allora R1 verra` richiesta all'istante (3+2)=5);
- la durata di tale richiesta.

3.3.2 Modalita' TextMode

La modalita' TextMode permette di creare un nuovo ambiente di simulazione scrivendo direttamente il file XML tramite un semplice editor testuale. Per avviare tale modalita', scegliere all'avvio del programma l'opzione "Nuova Simulazione" e quindi "Modalita' TextMode" oppure utilizzare gli appositi comandi sulla barra del menu' o sulla barra dei comandi; verra' dunque avviata l'esecuzione dell'editor testuale col quale creare il file XML.



- fig. 3.3.2 -

3.3.2.1 Funzionalita' dell'editor

L'editor offre le caratteristiche principali e minimali per una digitazione rapida e semplice adatta ad una vasta gamma di utenti. Le principali operazioni effettuabili si possono dividere in due categorie:

1. operazioni su file quali caricamento e salvataggio file XML;
2. operazioni di formattazione testo (taglia, copia, incolla e cancella testo selezionato, stampa modello XML, aggiungere tag di processo, risorsa, richiesta).

Tali operazioni sono richiamabili dalle barre dei comandi [vd. fig. 3.6b - 3.6c] o semplicemente cliccando con il tasto destro del mouse e selezionare l'operazione scelta da un menu' rapido.

Al fine di ottimizzare il tempo di scrittura e' possibile richiamare le operazioni di formattazione testo anche tramite le combinazioni tasti utilizzabili nei vari editor testuali dei sistemi operativi di maggior utilizzo.

L'utilizzo della modalita' TextMode non prevede la totale conoscenza del linguaggio XML definito per il software: e' possibile stampare sull'area di testo il modello XML tramite l'apposito tasto sulla barra dei comandi ed inserire i valori desiderati in maniera semplice e con bassa probabilita' di errore.

Per inserire il tag desiderato, posizionare il cursore in un punto dell'area di testo e selezionare una delle seguenti opzioni:

- *Modello completo XML:*inserisce un modello completo comprendente intestazione, una riga per la politica, per un processo, per una risorsa e per una richiesta;
- *Lista Politiche:* selezionare Politiche dal menu' Edit o dal menu' rapido cliccando col tasto destro, per avere una lista di tutte le politiche caricate dal software; selezionandone una viene inserito il tag corrispondente sull'area di testo;
- *Aggiungi Processo:* aggiunge un processo con i relativi campi dati e la priorita' se e'

stata selezionata, come descritto sopra, una politica che la richiede;
- *Aggiungi Risorsa:* aggiunge una risorsa con i relativi campi dati;
- *Aggiungi Richiesta:* aggiunge una richiesta con i relativi campi dati;

3.4 Carica Simulazione

Scegliendo "Carica Simulazione" si avvia la procedura di apertura di un file XML precedentemente salvato in memoria secondaria. Le modalita' di caricamento ambiente di simulazione sono due: modalita' grafica (VisualData) e modalita' testuale (TextMode).

3.4.1 Modalita' Grafica (VisualData)

Scegliendo di caricare una simulazione in modalita' grafica, i relativi dati vengono letti dal file e riportati in una struttura del tutto simile alla modalita' Wizard [cfr. §3.3.1], con pannelli per l'inserimento di politica, processi, eventuali risorse e a richieste. Le uniche differenze sono che la modalita' VisualData ha tutti i pannelli per l'inserimento dei dati contemporaneamente attivi, e non contiene quindi i bottoni per l'avanzamento sequenziale ("Avanti", "Indietro").

3.4.2 Modalita' Testuale

Scegliendo di caricare un ambiente di simulazione in modalita' TextMode e' possibile aprire e modificare il file XML corrispondente per essere quindi salvato o mandato in esecuzione. Per maggiori informazioni sull'editor testuale confrontare paragrafo §3.3.2.1

3.5 Esecuzione

Una volta creato l'ambiente di simulazione, e' possibile lanciare l'esecuzione cliccando sul tasto "Esegui". Se il file o le eventuali modifiche non sono stati salvati, l'applicazione richiede all'utente se desidera salvare la simulazione su file. Per lanciare l'esecuzione non e' necessario salvare i dati: se si sceglie di non salvarli, verra' creato un file temporaneo con i dati inseriti, che verra' automaticamente eliminato alla fine dell'esecuzione. A questo punto le finestre per l'inserimento dei dati vengono chiuse e si apre la sezione di esecuzione.

3.6 Barra dei comandi

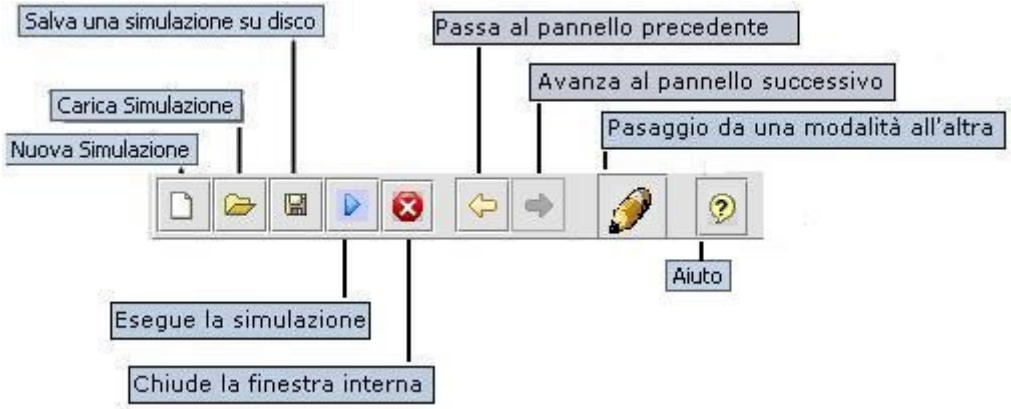
Sono utilizzabili la barra del menu' o le barre dei pulsanti [fig. 3.6x] per richiamare le operazioni principali per:
- gestione file;
- formattazione testo;
- visualizzazione dei dati.

I menu' e i comandi selezionabili sono i seguenti:

- **menu' File**:
  - *Nuova Simulazione*: per iniziare una nuova simulazione nella modalita' desiderata;
  - *Apri Simulazione*: per aprire una simulazione salvata nella modalita' desiderata;
  - *Nuovo...* : per iniziare una nuova simulazione nella modalita' corrente;
  - *Apri...* : per aprire una simulazione salvata nella modalita' corrente (se si e' in Wizard, il software passa in modalita' Visualdata [cfr. §3.4.1]);
  - *Salva*: salva una simulazione su file con estensione automatica xml;
  - *Salva con Nome...*: salva una simulazione su file con possibilita' di usare un altro nome;
  - *Chiudi*: chiude la simulazione corrente;
  - *Imposta cartella di caricamento politiche*: permette di selezionare la cartella contenente le politiche definite dall'utente per caricarle nel software;
  - *Esci*: chiude il software;
- **menu' Edit**:
  - *Esegui Simulazione*: manda in esecuzione la simulazione creata o caricata;
  - *Avanti (solo Wizard):* avanza al pannello successivo di inserimento dati;
  - *Indietro (solo Wizard):* indietreggia al pannello precedente di inserimento dati;
  - *Taglia (solo TextMode)*: taglia il testo selezionato;
  - *Copia (solo TextMode)*: copia il testo selezionato;
  - *Incolla (solo TextMode)*: incolla il testo tagliato o copiato;
  - *Cancella (solo TextMode)*: cancella il testo selezionato;
  - *Seleziona tutto (solo TextMode)*: seleziona l'intero testo del documento;
  - *Politiche (solo TextMode)*: elenco delle politiche caricate dal sistema;
- **menu' Simulazione**:
  - *Inizia*: lancia l'animazione dell'avanzamento del grafo temporale;
  - *Ferma*: ferma l'animazione del grafo temporale;
  - *Vai all'inizio*: riporta il grafo al punto iniziale;
  - *Vai alla fine*:porta il grafo al punto finale;
  - *Avanza*: avanza sul grafo di un passo;
  - *Indietreggia*: indietreggia sul grafo di un passo;
  - *Vai all'istante...*: avanza sul grafo fino al punto specificato dall'utente;
- **menu' Statistiche:**
  - *Visualizza...:* visualizza i dati delle statistiche per ogni processo quali:
    - Tempo medio/totale di Turn Around;
    - Tempo medio/totale di Attesa;
- **menu' Help**:
  - *Aiuto*: apre il manuale d'uso del software;
- **?** :
  - *Info...*: apre una finestra con l'elenco degli autori del software e il logo dell'azienda.

3.6.1 Barra modalita' Wizard

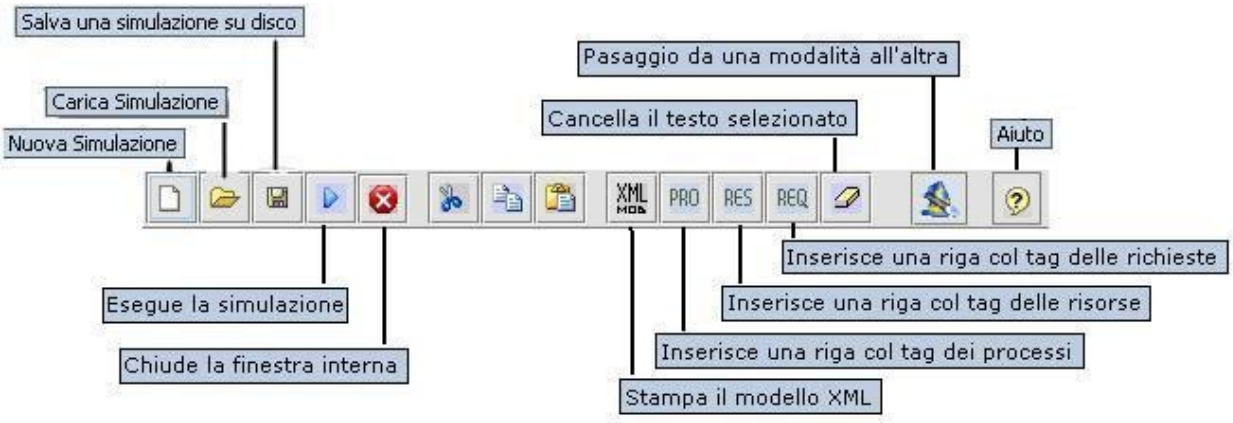
Creando o caricando una simulazione in modalita' grafica si puo' utilizzare la seguente barra del menu':



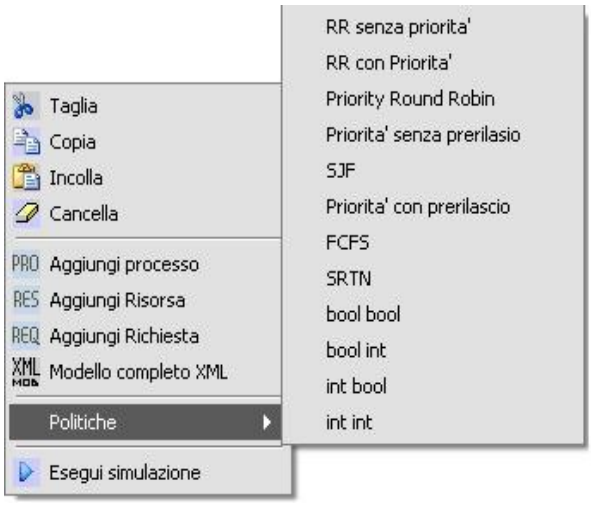
- fig. 3.6a -

3.6.2 Barra modalita' TextMode

Creando o caricando una simulazione in modalita' testuale si puo' utilizzare la seguente barra del menu' [fig. 3.6b] ed il menu' a scelta rapida [fig. 3.6c] richiamabile con il tasto destro del mouse:



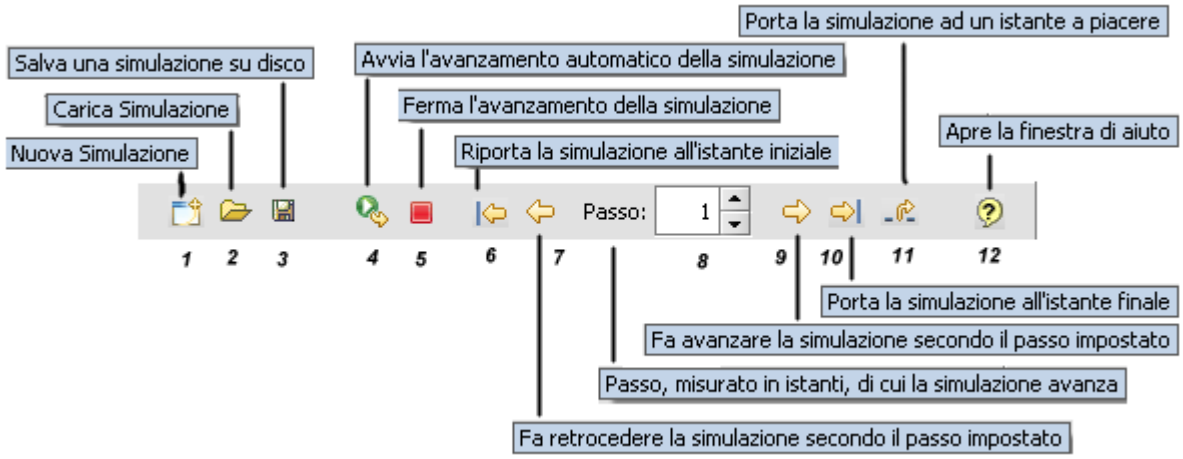
- fig. 3.6b -



- fig. 3.6c -

3.6.3 Barra di esecuzione

Quando viene mandata in esecuzione, e' possibile gestire la simulazione e le animazioni con la seguente barra del menu':



- fig. 3.6d -

3.7 Gestione Simulazione

Tramite il pannello di controllo utente e' possibile visualizzare la simulazione in 3 modi diversi:

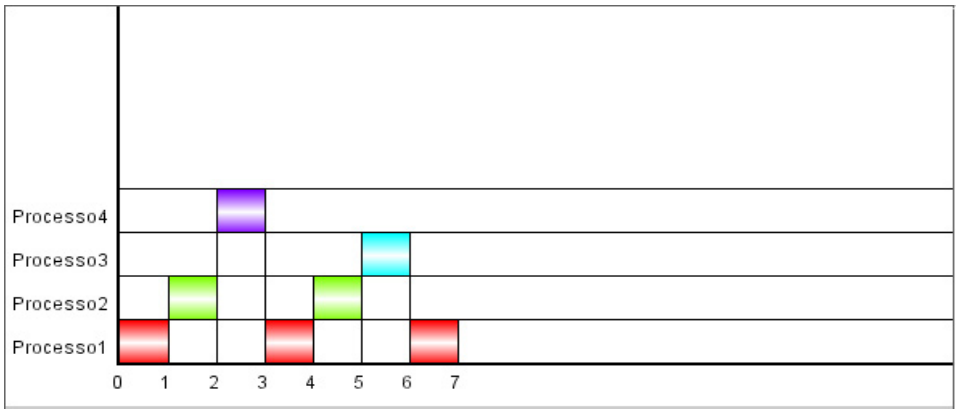
- "Passo a Passo": l'utente esegue la simulazione avanzando (bottone 9) o indietreggiando (bottone 7). E' possibile scegliere il passo della simulazione t (box 8)
- "Automatica": La simulazione avanza automaticamente dall'inizio alla fine (bottone 4). Come sopra e' possibile scegliere il passo della simulazione. E' tuttavia possibile bloccare la simulazione (bottone 5) e riprendere in modalita' "Automatica" o "Passo a Passo".

- "Finale": mostra l'ultimo istante della simulazione. Utile per visualizzare la completa esecuzione dei processi o visualizzare le statistiche finali

La simulazione puo' essere gestita anche utilizzando i comandi del menu' "Simulazione"[cfr. §3.6]

3.7.1 Avanzamento processi

Consiste in un grafico che visualizza la schedulazione dei processi, nell'asse delle ascisse sono enumerati gli istanti, in quella delle ordinate il nome dei processi (nel caso il nome sia troppo lungo verra' troncato). Ogni spazio colorato determina il processo in esecuzione in quel determinato istante.
Per una piu' rapida visualizzazione del grafico ad ogni processo e' stato assegnato un particolare colore.



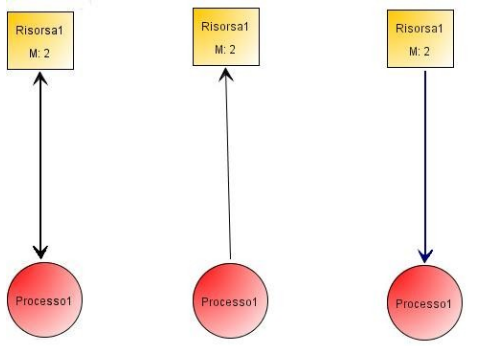
- fig. 3.7.1 -

3.7.2 Assegnazione risorse

Consiste in un grafico rappresentante le richieste delle risorse da parte dei processi e l'eventuale utilizzo di questi.
Le risorse sono rappresentate da dei rettangoli contenenti il nome della risorsa e la sua molteplicita', i processi invece da dei cerchi contenenti il nome del processo.

Processi e risorse sono collegati tra loro da tre tipi differenti di frecce che identificano tre situazioni distinte:

1. Nello stesso istante il processo "p" effettua la richiesta della risorsa "r" e la ottiene.
2. Il processo "p" richiede la risorsa "r"
3. La risorsa "r" è assegnata al processo "p"



1) 2) 3)

3.7.3 Coda dei Processi

Sono presenti due liste in cui è visualizzata la coda ordinata dei processi in stato di Pronto e la coda ordinata dei processi in stato di Attesa.

3.7.4 Simulazione Testuale

E' possibile visionare la Simulazione in formato testuale.

Per ogni istante viene indicato:
- il processo in esecuzione;
- la Lista dei processi in stato di pronto;
- la Lista dei processi in stato d'attesa;
- la Lista dei processi terminati;
- le eventuali assegnazioni delle risorse a processi;
- le eventuali richieste di risorse da parte di processi;

3.7.5 Statistiche

Dal Menu "Statistiche" alla voce "Visualizza..." è possibile aprire la finestra in cui vengono visualizzate le statistiche della simulazione.

Sono indicate statistiche della simulazione nel suo complesso:
- Tempo Totale di Turn Around;
- Tempo Medio di Turn Around;
- Tempo Totale di Attesa;
- Tempo Medio di Attesa;

Le statistiche specifiche per ciascun processo sono:
- Tempo di Turn Around;
- Tempo di Attesa;
- Risorse utilizzate;

3.8 Salvare la simulazione

E' possibile salvare i dati relativi l'ambiente di simulazione utilizzando gli appositi pulsanti sulla barra dei comandi o direttamente dal menu' principale [vd. fig 3.6x]

3.8.1 Salvare l'ambiente di simulazione

I dati relativi l'ambiente vengono salvati su file XML con estensione XML che puo' essere caricato nelle due modalita' previste [cfr. §3.4].

E' possibile salvare i dati inseriti in qualsiasi momento, anche se l'inserimento dei dati della simulazione e' errato o non completo.

3.8.2 Salvare i risultati dell'esecuzione

E' possibile salvare i grafici e le statistiche esportandoli in formato XML.

4. TEORIA DEI PROCESSI

Un processo è un programma in esecuzione, il modo più semplice per capire il concetto di processo è pensare ad un sistema a quanto di tempo. Periodicamente il sistema operativo decide di eseguire un processo differente, quindi l'alternarsi dei processi (dei programmi) nell'uso del processore da l'illusione all'utente del parallelismo, quindi di effettuare contemporaneamente più azioni. Processi indipendenti possono avanzare concorrentemente senza alcun vincolo di ordinamento, mentre se condividono risorse è necessaria l'introduzione di meccanismi di sincronizzazione di accesso. La modalita' di accesso indivisa ad una variabile condivisa viene detta in mutua esclusione, cioè la variabile viene usata solamente dal processo che la possiede e l'accesso viene inibito a qualsiasi altro processo tramite una variabile definita lucchetto "lock" o mutual exclusion "mutex".

4.1 Comunicazione e sincronizzazione tra processi

4.1.1 Monitor

L'uso dei semafori a livello di programma è difficile e rischioso poiché può causare situazioni di blocco infinito "deadlock" e situazioni erronee di difficile verifica. Linguaggi evoluti e di alto livello offrono all'utente strutture esplicite di controllo delle regioni critiche dette appunto monitor. Il monitor individua le regioni critiche e il compilatore automaticamente inserisce il codice necessario al controllo degli accessi; il monitor è in effetti un aggregato di sottoprogrammi e strutture dati che possono accedere alle variabili interne e permette a solo un programma per volta di essere attivo al suo interno. Questa proprieta' è garantita dai meccanismi del supporto a tempo di esecuzione del linguaggio di programma corrente, il cui

codice è inserito dal compilatore nel programma eseguibile.
La sola garanzia di mutua esclusione può non bastare ad affrontare il problema, si procede pertanto all'introduzione di due procedure operanti su variabili speciali dette condition variables che consentono di modellare condizioni logiche specifiche per il problema:

- Wait – viene forzata l'attesa per il chiamante
- Signal – risveglia il processo in attesa

4.1.2 Barriere

Le barriere consentono di sincronizzare gruppi di processi in quanto le attivita' cooperative sono suddivise in fasi
ordinate. La barriera blocca tutti i processi che la raggiungono fino all'arrivo dell'ultimo.

4.2 Deadlock(stallo):

In molte applicazioni un processo può avere bisogno di accedere in modo esclusivo non ad una sola risorsa bensi' a molte. Supponiamo che un processo A necessiti della risorsa r1 e la ottiene, analogamente un processo B necessita' e ottiene la risorsa r2. A questo punto A chiede l'uso della risorsa r2 ma la otterra' solo quando verra' rilasciata da B, sfortunatamente B chiede l'uso di r1 che è gia' in uso da A,a questo punto i processi entrano in attesa per un tempo infinito. Una situazione di questo tipo è detta deadlock.
Per evitare lo stallo esistono almeno tre strategie:

1. Prevenzione
2. Riconoscimento e recupero
3. Indifferenza

4.2.1 Prevenzione

Si impediscono le condizioni ritenute critiche per il sistema. Si considerano in maniera accurata gli accessi esclusivi ad una risorsa, si evitano gli accumuli di risorse, cosa che però è molto difficile da eliminare, le inibizioni del prerilascio e l'attesa circolare.
La prevenzione si può fare ad ogni richiesta di accesso, verificando se questo può portare ad uno stallo, ma in questo caso la verifica è un onore pesante e in caso affermativo non è ben chiaro cosa bisogna fare. Si può anche chiedere preventivamente ad ogni processo quali risorse impieghera' in modo da ordinarli in maniera conveniente.

4.2.2 Riconoscimento e recupero

Si ammette e si tollera il verificarsi di uno stallo avendo però la capacita' di invocare una procedura di recupero.

4.2.3 Indifferenza

Si considera molto bassa la possibilita' di uno stallo e non si prende nessuna precauzione contro il suo verificarsi.

5. POLITICHE DI ORDINAMENTO PROCESSI

5.1 Programmazione concorrente

Per loro natura molti sistemi sono concorrenti e in questo caso bisogna dotarsi di strumenti per rappresentare e rendere esplicito il loro parallelismo potenziale. La programmazione concorrente è l'insieme di notazioni tecniche usate per esprimere il parallelismo insito in un problema mediante la rappresentazione dei singoli flussi di controllo e per risolvere i problemi di comunicazione da esso derivante.
Un programma concorrente corretto non richiede di specificare l'esatto ordine di esecuzione dei processi, poiché l'ordinamento locale tra gruppi di processi viene reso tramite primitive di comunicazione e sincronizzazione, e quest'ultimo non deve aver alcuna influenza sul risultato finale.

5.2 Classificazione dei sistemi

Diverse classi di applicazioni richiedono politiche di ordinamento per processi diverse, e in generale sono presenti
tre classi generali:

- *A lotti:* Ordinamento precostituito; lavori di lunga durata e limitata urgenza dove il prerilascio non è necessario; le caratteristiche desiderabili sono:
  - Massimizzazione del troughput;
  - Brevita' del turnaround time;
  - Massimo utilizzo delle risorse di calcolo.
- *Sistemi interattivi:* Grande varieta' di attivita'. In questo caso il prerilascio è essenziale. Le caratteristiche desiderabili sono:
  - Rapidita' di risposta;
  - Soddisfazione delle aspettative degli utenti.

Caratteristiche desiderabili un tutte le politiche di ordinamento:

1. Fairness, cioè equita' in tutte le oppurtunita' di esecuzione;
2. Enforcement, cioè coerenza nell'applicazione della politica in tutti i processi;
3. Bilanciamento nell'uso di tutte le risorse del sistema.

5.3 Politiche di ordinamento

5.3.1 Per sistemi a lotti

- **FCFS:** First come first served. Senza prerilascio e senza priorita'. L'ordine di esecuzione è uguale all'ordine di arrivo. Caratterizzato da massima semplicita' e basso utilizzo di risorse.
- **SJB**: Shortest job first. Senza prerilascio e richiede la conoscenza dei tempi richiesti di esecuzione. Esegue prima il lavoro più breve e non è equo con i lavori non presenti all'inizio.
- **SRTN:** Shortest remaining time next. Aggiunge il prerilascio a SJB. Esegue prima il processo più veloce a completare e tiene conto dei nuovi processi che arrivano all'esecuzione.

5.3.2 Per sistemi Interattivi

- *OQ:* Ordinamento a quanti. Chiamato anche "round robin" è con prerilascio ma senza priorita'. Ogni processo esegue al più un quanto per volta ed è presente una lista circolare dei processi.
- *OQP:* Ordinamento a quanti con priorita'. I quanti dipendono in questo caso dal livello di priorita' del processo.
- *GP:* Con garanzia per processo. Con prerilascio e con promessa di una data quantita' di tempo di esecuzione, tenendo presente che le necessita' di ciascun processo devono essere conosciute, o almeno stimate a priori. Viene eseguito il lavoro maggiormente penalizzato rispetto alla promessa di esecuzione.

- *SG:* Senza garanzia. Ci sono prerilascio e priorita'. Opera sul principio della lotteria:
  1. Ogni processo riceve un numero da giocare;
  2. A priorita' più alte corrispondono più numeri da giocare;
  3. Ad ogni scelta per assegnazione di risorsa, essa va al processo possessore del numero estratto;
  4. Le estrazioni avvengono periodicamente o ad eventi;
  5. Il comportamento è impredicibile sul breve periodo, ma tende a stabilizzarsi statisticamente nel tempo;
- *GU:* Con garanzia utente. Come GP ma con garanzia riferita a ciascun utente.

Le politiche sopracitate sono da ritenersi utili per fornire all'acquirente una infarinatura sull'argomento, e potrebbero non essere presenti nel software al momento dell'acquisto. Si ricorda che possono essere aggiunte tramite le funzionalita' offerte dal prodotto.

5.4 Inversione di priorita'

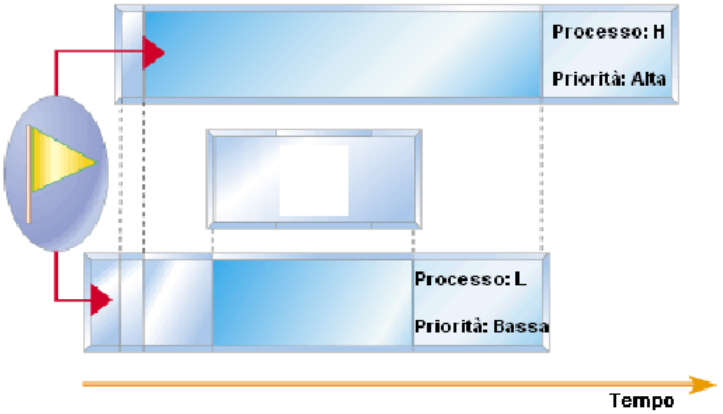
I sistemi operativi utilizzano uno scheduler con prerilascio basato su priorita'. Questi sistemi assegnano ad ogni processo un unico livello di priorita'. Lo scheduler si accerta che di tutti i processi pronti a funzionare, sia in esecuzione sempre il processo con priorita' maggiore. Per raggiungere questo obiettivo, lo scheduler può rilasciare un processo di priorita' bassa durante la sua esecuzione. Poiché i processi utilizzano le risorse, eventi fuori dal controllo dello scheduler possono impedire al processo pronto con priorita' maggiore di andare in esecuzione quando dovrebbe. Se questo accade, una scadenza critica potrebbe essere mancata, inducendo il fallimento del sistema. Si dice che avviene un'inversione di priorita' quando il processo pronto a priorita' maggiore non riesce ad entrare in esecuzione quando dovrebbe.

5.4.1 Condivisione di risorse

In qualunque momento due processi condividono una risorsa in un sistema che impiega uno scheduler basato su priorita', uno di loro avra' solitamente priorita' maggiore dell'altro. Il processo con priorita' maggiore si aspetta di andare in esecuzione non appena è pronto. Tuttavia, se il processo con priorita' minore sta usando la risorsa condivisa quando il processo a priorita' maggiore diventa pronto, quest'ultimo processo deve aspettare che il processo in esecuzione finisca la sua esecuzione. Diciamo che il processo a priorita' maggiore è in attesa bloccante della risorsa.

5.4.2 Il problema

La difficolta' si presenta a tempo di esecuzione, quando un processo con priorita' media causa il prerilascio di un processo con priorita' bassa attraverso la richiesta di utilizzo di una risorsa condivisa su cui è in attesa il processo con priorita' alta. Al contrario, se il processo con priorita' alta è pronto ad andare in esecuzione ma un processo con priorita' media è attualmente in esecuzione, si dice che avviene un'inversione di priorita'.



- fig. 5.4 -

Una sequenza critica è illustrata nella figura precedente. Il processo a bassa priorita' L e quello ad alta priorita' H condividono una risorsa. Dopo che il processo L ottiene la risorsa, il processo H diventa pronto. Tuttavia il processo H deve aspettare che L termini di utilizzare la risorsa. Prima che L finisca di utilizzare la risorsa, il processo M a media priorita' diventa pronto, causando il prerilascio di L. Mentre è in esecuzione il processo M, il processo H è costretto a rimanere in attesa.

5.4.3 Soluzioni

La ricerca sull'inversione di priorita' ha portato a due possibili soluzioni.
Il primo è detto "priority inheritance". Questa tecnica consiste nel fatto che un processo con priorita' bassa eredita la priorita' del processo con priorita' maggiore in attesa della risorsa condivisa. Questo cambiamento di priorita' dovrebbe avvenire non appena il processo con priorita' maggiore entra in attesa della risorsa; dovrebbe concludersi quando la risorsa è liberata. Ciò richiede l'intervento del sistema operativo.
La seconda soluzione, "priority ceilings", associa una priorita' ad ogni risorsa; lo scheduler poi trasferisce tale priorita' ad ogni processo che accede alla risorsa. La priorita' assegnata alla risorsa è la priorita' del relativo processo a priorita' maggiore, più uno. Una volta che un processo ha terminato di utilizzare la risorsa, la sua priorita' torna ad essere quella precedente.

6. AGGIUNTA NUOVE POLITICHE DI SCHEDULAZIONE

L'applicazione S.G.P.E.M. v2 permette all'utente ESPERTO di aggiungere nuove politiche di schedulazione senza intervenire in alcun modo sull'architettura del sistema.

Le nuove politiche sono realizzate tramite classi java che devono essere create nella

directory che l'utente sceglie per il caricamento dinamico di politiche utente ("data/policy/" di default).

6.1 Creazione della classe

Per realizzare una classe di politica di schedulazione bisogna implementare l'interfaccia sgpem.kernel.policy.Policy; a tale scopo si consiglia di aggiungere al proprio classpath il file sgpem2.jar per importare l'interfaccia Policy e l'eccezione NoprocessExecuted con le seguenti istruzioni:

- import sgpem2.kernel.policy.Policy;
- import sgpem2.kernel.NoProcessExecuted;

N.B: La classe da realizzare non deve contenere dichiarazione di appartenenza a package.

6.2 L'interfaccia Policy

L'interfaccia da implementare e' la seguente, in rosso sono riportati i commenti ai metodi e campi dati:

```
/*
File: Policy.java
Versione: 1.0
Autore: Scattolin Mattia
Data di creazione: 27/06/2006
Data ultima modifica: 27/06/2006
Diario delle modifiche:
v1.0: adattamento della classe e riscrittura dei commenti

Condizioni di utilizzo:
This file is part of S.G.P.E.M.2

S.G.P.E.M.2 is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

S.G.P.E.M.2 is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with S.G.P.E.M.2; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
*/

package sgpem2.kernel.policy;

import java.util.Vector;
import sgpem2.kernel.NoProcessExecutedException;
```

```
/**
* Interfaccia che specifica le politiche di schedulazione: solo le classi che
* la implementano saranno riconosciute come nuove politiche dall'applicazione.
*
* ATTENZIONE: Le classi implementanti questa interfaccia non possono avere
* costruttori fatta eccezione per quello senza parametri.
*/
public interface Policy {

// Dati statici
/**
* Nome esteso della politica (es.: "First come, first served").
*/
public static final String name=new String("");

/**
* Sigla identificativa della politica (es.: "FCFS").
*/
public static final String short_name=new String("");

/**
* Nome della politica per il file xml (es.: "FirstComeFirstServed").
* N.B.: Per ragioni di efficienza tale nome deve coincidere con il nome
* del file della politica (esclusa l'estensione .java).
*/
public static final String xml_name=new String("");

/**
* Array contenente la descrizione di eventuali parametri relativi alla
* politica di schedulazione (es.: {"Time slice", "Priorita' crescente"}).
*/
public static final String[] arg_description={};

/**
* Array contenente la stringa xml di eventuali parametri relativi alla
* politica di schedulazione (es.: {"time_slice", "crescent_priority}).
* N.B.: Tale array deve avere la stessa dimensione del precedente.
*/
public static final String[] arg_xml={};

/**
* Array contenente il tipo di eventuali parametri relativi alla
* politica di schedulazione (es.: {Integer.class, Boolean.class}).
* N.B.: Anche questo array deve avere la stessa dimensione dei precedenti.
*/
public static final Class[] arg_type={};

/**
* Array contenente il valore di eventuali parametri relativi alla
* politica di schedulazione.
* Si possono dare dei valori di default o lasciare degli eleminti
* esplicitamente a null (in tal caso bisognera' invocare il metodo
* initializeParameters per dar loro un valore.
* (es.: { null, Boolean.FALSE }).
```

```
*/
public Object[] arg_value={};

/**
* Indica se la politica di schedulazione tiene conto della priorita'
* dei processi.
*/
public static final boolean use_priority=false;


// Metodi
/**
* Restituisce il nome esteso della politica.
*/
public abstract String getName();

/**
* Restituisce il nome in forma abbreviata della politica.
*/
public abstract String getShortName();

/**
* Restituisce il nome della politica da usare nel file xml.
*/
public abstract String getXmlName();

/**
* Restituisce la descrizione di eventuali parametri della politica.
*/
public abstract String[] getArgDescription();

/**
* Restituisce la stringa xml che definisce eventuali parametri della
* politica.
*/
public abstract String[] getArgXmlName();

/**
* Restituisce il tipo di eventuali parametri della politica.
*/
public abstract Class[] getArgType();

/**
* Restituisce il valore di eventuali parametri. I parametri non ancora
* inizializzati che non hanno valore di default sono uguali a null.
*/
public abstract Object[] getArgValue();

/**
* Inizializza un parametro della politica tramite il suo indice
* nell'array.
* N.B.: Ovviamente il tipo dinamico dell'oggetto parameterValue deve
* coincidere con quello del relativo oggetto Class nell'array ritornato
* dal metodo getParameterType().
```

```
*
* @throws InvalidParameterException Se il valore del parametro non e'
* del tipo giusto o comunque e' non ammissibile. Invocare getMessage()
* sull'eccezzione per i dettagli.
*/
public abstract void initializeParameter(Object parameterValue, int index)
throws InvalidParameterException;

/**
* Indica se la politica utilizza la priorita'.
*/
public abstract boolean usePriority();

/**
* E' il metodo che implementa effettivamente la politica di schedulazione:
* ricevendo la lista dei processi da ordinare, l'id dell'ultimo processo
* andato in esecuzione (-1 se nessun processo ha eseguito) e l'id del
* processo appena arrivato allo stato di pronto (-1 se in quest'istante
* non c'e' nessun nuovo processo).
* Il sistema invoca in ogni istante della simulazione questo metodo con
* i dati aggiornati ed esegue il processo che si trova in prima posizione
* dopo l'invocazione.
* I processi da ordinare sono rappresentati da array di interi che ne
* identificano le caratteristiche fondamentali cioe':
* { id, istante_di_arrivo, tempo_totale_di_esecuzione,
* tempo_rimanente_di_esecuzione, priorita' }.
*
* @return un Vector con gli stessi processi ricevuti
* come input dopo averne eventualmente modificato l'ordine.
* @throws NoProcessExecutedException Per segnalare al kernel che vi sono
* stati dei problemi in fase di ordinamento processi.
*/
public abstract Vector sortList(Vector processList, int lastExecutedId, Vector newProcesses)
throws NoProcessExecutedException;

}
```

Vedere il file "data/policy/ShortestRemainingTimeNext.java" inserito tra le politiche utente per fornire un'esempio di realizzazione di una nuova politica.

6.3 Compilazione della classe

Dopo aver creato la classe e aver aggiunto al classpath di sistema il file sgpem2.jar, compilarlo semplicemente con "javac NomeClasse.java" e avviare l'applicazione. Per verificare l'avvenuto inserimento della politica, controllare nell'elenco delle politiche nella barra del menu' dalla modalita' TextMode [cfr. §3.6] oppure dalla lista delle politiche selezionabili dall'apposita combobox in modalita' Wizard [cfr §3.3.1.1].