PLANIFICATOR PROCESE

Manual de utilizare

realizat de
Vlad Manea
Sebastian Dițu
Codruț Burdujoc

CUPRINS

ARGUMENT

În contextul în care, în secolul al XXI-lea, accesul la informație nu mai e o problemă, adevăratul obstacol îl constituie filtrarea şi, în special, asimilarea corectă a informațiilor.

Deşi sursele de documentare privind sistemele de operare, în general, şi planificarea proceselor, în special, sunt diverse, acestea sunt însoțite de cele mai multe ori doar de câteva imagini orientate pe o sarcină dată.

Studenții sunt puşi în fața situației de a stabili conexiuni între ele însă, de cele mai multe ori, acest proces este dificil şi mare cauzator de confuzii şi improvizații. Pentru a depăşi aceste probleme, am realizat un proiect care prezintă informațiile într-un mod interactiv.

Simulatorul construit realizează planificarea job-urilor la procesor utilizând strategiile actuale de planificare. El oferă posibilitatea configurării strategiilor prin datele de intrare la pornirea programului.

Proiectul constituie un bun suport de curs, dar poate fi folosit şi ca resursă proprie de studiu, cu mențiunea de a consulta şi alte surse de informație din acest domeniu.

UTILIZARE

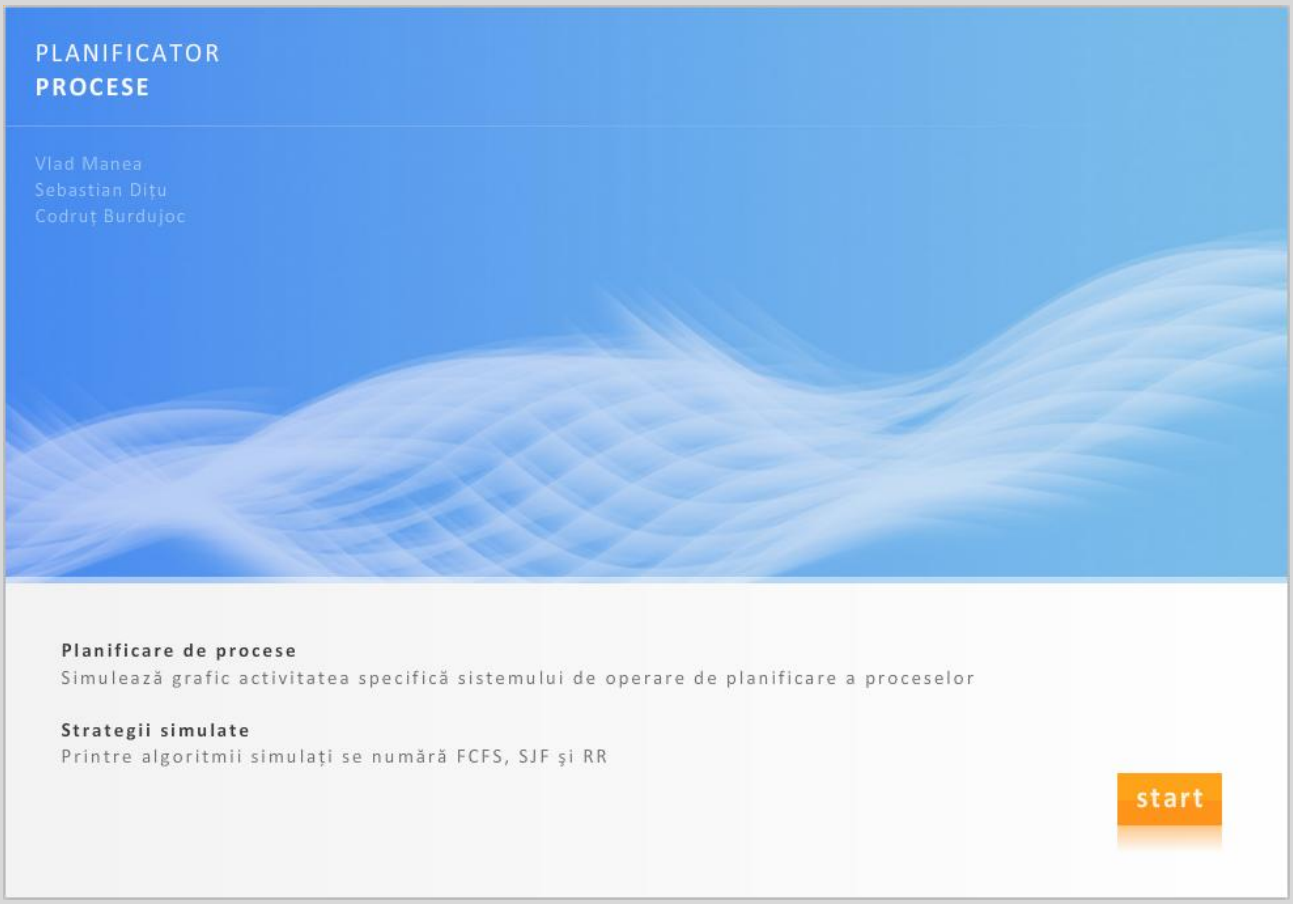
Cerințe sistem

Pentru a rula aplicația în condiții optime, sunt necesare:

- Sistem de operare Linux sau Windows
- Flash Player instalat, minim versiunea 9

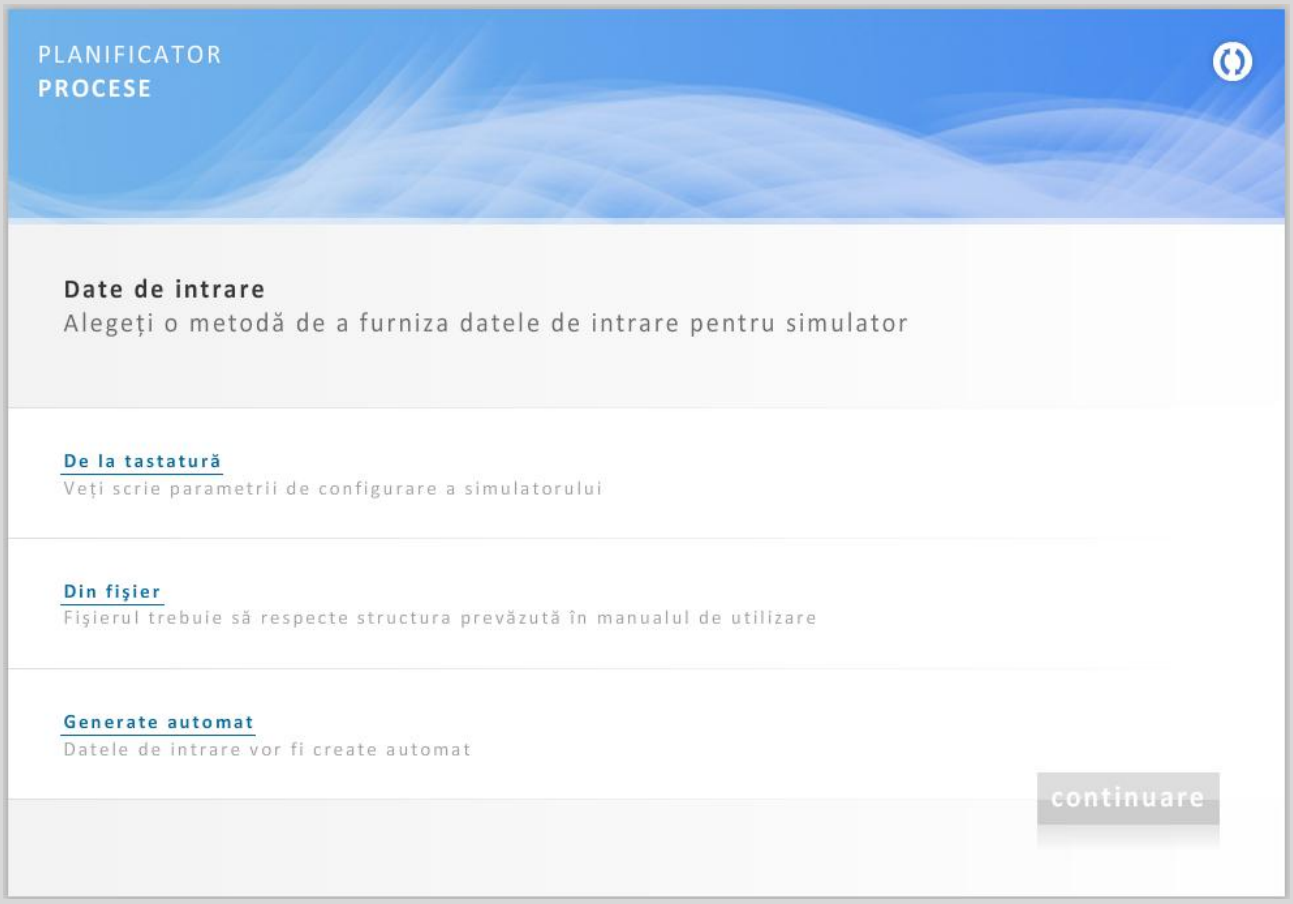
Interfață

Aplicația propune o interfață simplă şi uşor de utilizat. Am înlăturat din ea informațiile redundante pentru a îmbunătăți experiența utilizatorului.



Date de intrare

Utilizatorul este invitat să aleagă o metodă prin care să furnizeze datele de intrare. Opțiunile disponibile sunt:

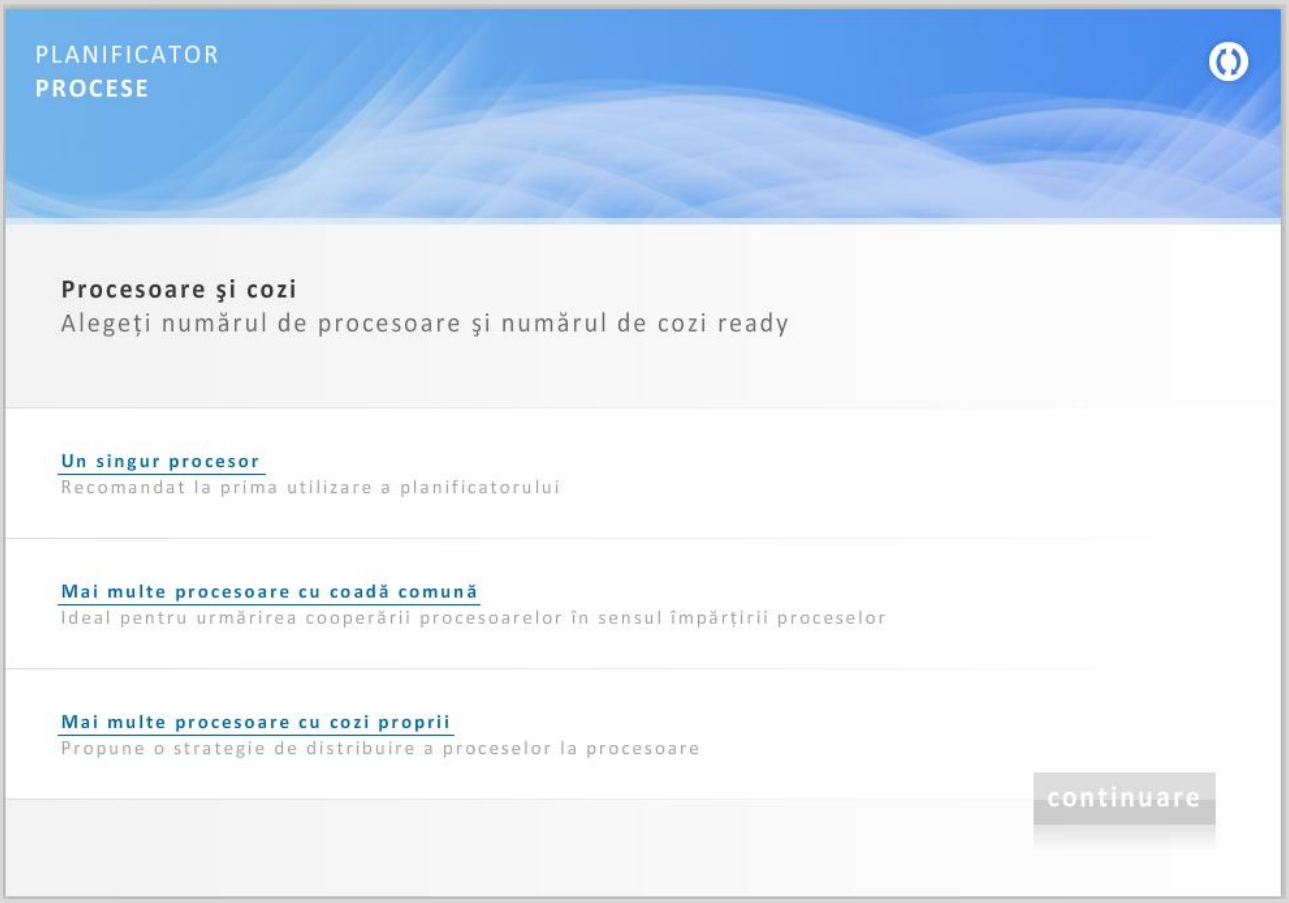


De la tastatură

În cazul în care se optează pentru această opțiune, utilizatorul va alege contextul în care se va desfăşura simularea. La fiecare pas, este posibilă întoarcerea la început prin intermediul butonului înapoi, situat în partea superioară dreaptă.

Procesoare şi cozi

În acest pas, utilizatorul poate specifica tipul sistemului pe care se va face simularea şi anume: un singur procesor sau mai multe (cu coadă comună sau cu cozi separate).



În funcție de opțiunea aleasă, se va cere configurarea unor parametri specifici. De exemplu, dacă sunt alese mai multe procesoare cu o coadă comună, se va configura doar acea coadă.

Dacă, însă, se optează pentru mai multe procesoare cu cozi proprii, se va configura fiecare coadă în parte.

Strategii de planificare

În cazul în care s-a ales o opțiune cu mai multe procesoare, se specifică numărul lor, cel mult 3. Dacă se specifică o valoare diferită de 2 sau 3, câmpul ia o valoare implicită şi anume 2. Această regulă, a resetării la valoarea implicită în cazul furnizării unei date diferită de cele admise, este valabilă pentru toate câmpurile care pot fi completate de utilizator.

Pentru fiecare coadă se va alege o strategie de planificare a proceselor:

- **First Come First Served (FCFS)** prevede ca procesele să fie adăugate în coadă şi preluate de procesor în ordinea sosirii lor.

- **Shortest Job First (SJF)** planifică procesele astfel încât cele care au timpul de serviciu mai scurt să aibe prioritate.
    - Această strategie poate fi una preemptivă, în sensul că apariția unui proces mai bun în coadă determină întreruperea procesului aflat în procesor la acel moment. Pentru această opțiune, se va alege **Preempție forțată**.

- **Priorități** selectează procesele în ordinea priorităților lor.
    - Şi această strategie poate fi preemptivă: un proces cu prioritatea mai mare aflat în coada asociată unui procesor va determina scoaterea procesului din respectivul procesor în cazul în care acesta există. Opțiunea se activează cu **Preempție forțată**.
    - Prioritatea unui proces aflat într-o coadă ready creşte proporțional cu timpul petrecut de acel proces în coadă. Pentru această opțiune se alege **Priorități dinamice**.

- **Round Robin (RR)** cere în prealabil un număr maxim de unități de timp în care poate fi rulat un proces de către un procesor (cuanta). Cuanta poate avea valori între 1-5. Valoare implicită a cuantei este 1. În cazul în care un proces are nevoie de mai multe unități de timp decât valoarea cuantei, el va fi scos din procesor.

- **Round Robin (RR) cu priorități** are în plus proprietatea că procesele sunt ordonate în coada ready după prioritățile lor. Din acest motiv, această strategie permite aceleaşi facilități ca **Priorități**:
    - **Preempție forțată**
    - **Priorități dinamice**

  Şi în acest caz, trebuie specificată cuanta.

Procese

La acest pas, utilizatorul va specifica numărul de procese (între 2-9, cu valoare implicită 2). Pentru fiecare proces în parte, se vor completa:

- **Momentul de intrare** reprezintă unitatea de timp în care procesul este lansat în execuție şi plasat într-o coadă ready. Strategia de selecție a cozii ready pentru un proces în această situație este de a alege coada ocupată cu cel mai mic număr de procese. Intervalul de valori pentru momentul de intrare este 0-9, iar valoarea implicită e 0.
- **Prioritatea implicită** este prioritatea cu care intră procesul în coada ready, în cazul în care aceasta are o strategie cu priorități. Intervalul posibil este 0-9, iar valoarea implicită este 0.
- **Prioritatea dinamică** reprezintă numărul cu care creşte prioritatea implicită a unui proces la trecerea unei unități de timp, dacă acesta se găseşte într-o coadă cu o asemenea strategie. Intervalul posibil este 0-9, iar valoarea implicită este 0.
- **Perioadele de activitate** reprezintă secvențele de instrucțiuni procesor, respectiv intrare/ieşire. Numărul acestora trebuie să fie un

impar între 1-9, iar numărul implicit este 1. Imparitatea este necesară pentru formatul:

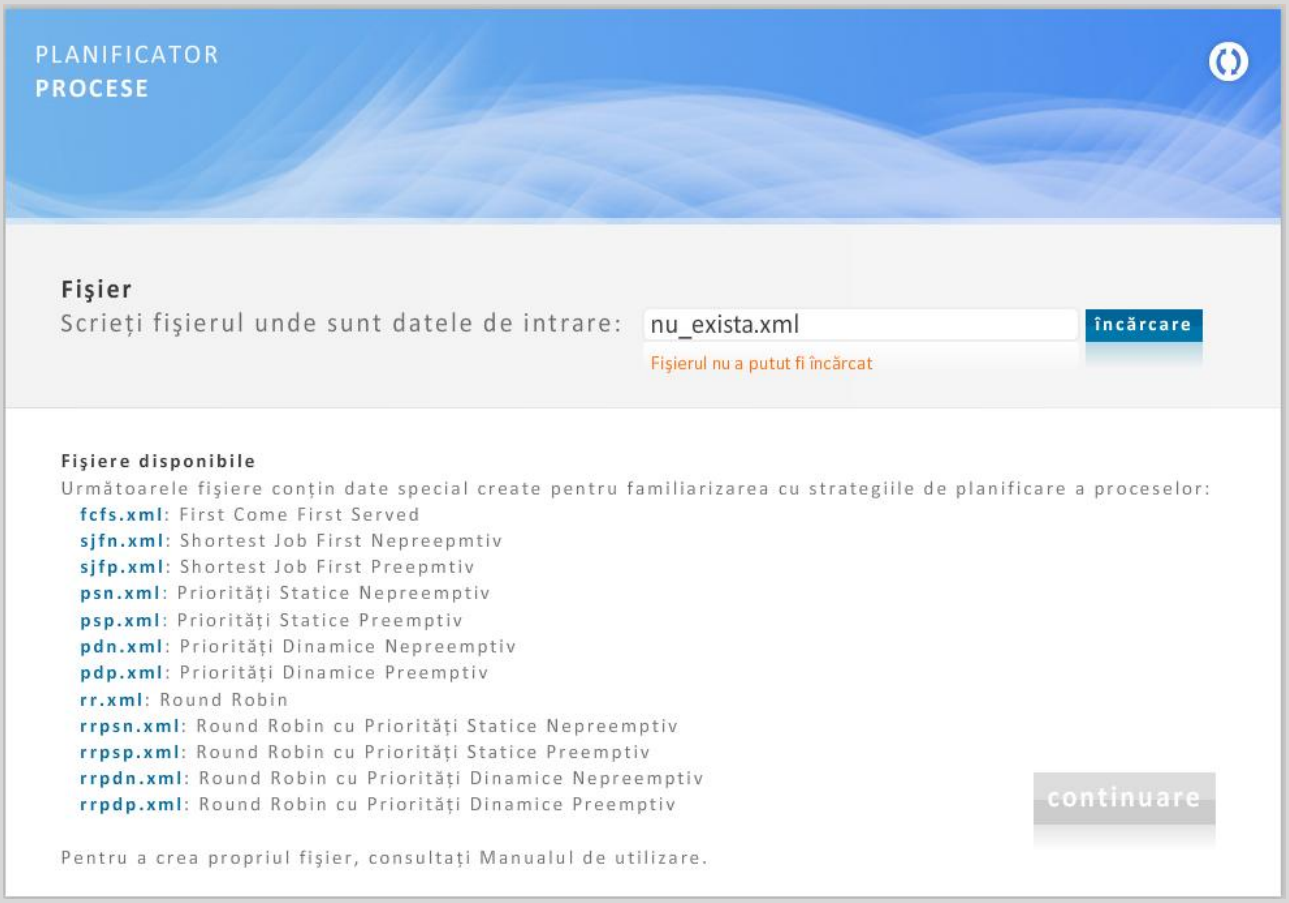
- Secvența 1, de instrucțiuni procesor
- Secvența 2, de instrucțiuni intrare/ieşire
- Secvența 3, de instrucțiuni procesor

  ...
- Secvența 2k-1, de instrucțiuni procesor
- Secvența 2k, de instrucțiuni intrare/ieşire
- Secvența 2k+1, de instrucțiuni procesor

Timpii necesari pentru aceste secvențe pot lua valori între 1-5, iar timpul implicit este 1.

Din fișier

Datele de intrare pot fi citite și prin intermediul unui fișier .xml. Acesta conține aceleași date de intrare pe care utilizatorul le-ar fi specificat de la tastatură. Utilizatorul trebuie să scrie numele fișierului, să apese pe butonul **încărcare**. Dacă fișierul conține date valide se activează butonul **continuare**.



Fișiere predefinite

Pentru ușurarea învățării strategiilor de planificare a proceselor pe care le-am implementat, am prevăzut proiectul cu un set de fișiere .xml cu aceleaşi date pe cozi cu fiecare strategie în parte:

- ✓ **fcfs.xml**: First Come First Served
- ✓ **sjfn.xml**: Shortest Job First Nepreemptiv
- ✓ **sjfp.xml**: Shortest Job First Preemptiv
- ✓ **psn.xml**: Priorități Statice Nepreemptiv
- ✓ **psp.xml**: Priorități Statice Preemptiv
- ✓ **pdn.xml**: Priorități Dinamice Nepreemptiv
- ✓ **pdp.xml**: Priorități Dinamice Preemptiv
- ✓ **rr.xml**: Round Robin
- ✓ **rrpsn.xml**: Round Robin cu Priorități Statice Nepreemptiv
- ✓ **rrpsp.xml**: Round Robin cu Priorități Statice Preemptiv
- ✓ **rrpdn.xml**: Round Robin cu Priorități Dinamice Nepreemptiv
- ✓ **rrpdp.xml**: Round Robin cu Priorități Dinamice Preemptiv

Structura unui fișier

Un fişier de intrare cu date considerate valide trebuie să respecte o structură clară. Următorul fişier .xml este valid.

```
<date>
 <cozi>
  <coada strategie="Round Robin (RR) cu priorități" cuanta="1"
    preemptiefortata="da" prioritatidinamice="da" />
  <coada strategie="Priorități" prioritatidinamice="da" />
  <coada strategie="First Come First Served (FCFS)" />
 </cozi>
 <procesoare>
  <procesor cuanta="7" />
  <procesor />
  <procesor />
 </procesoare>
 <procese>
  <proces momentintrare="0" prioritateimplicita="7"
    prioritatedinamica="2">
   <perioada timp="4" />
  </proces>
  <proces momentintrare="2" prioritateimplicita="3"
    prioritatedinamica="4">
   <perioada timp="5" />
   <perioada timp="2" />
   <perioada timp="3" />
  </proces>
 </procese>
</date>
```

Cerințele generale pentru un fişier .xml sunt:

- Să conțină un tag cu numele date: *<date>*.
- Să conțină în tagul *<date>*, în această ordine, tagurile *<cozi>*, *<procesoare>*, *<procese>*.
- În tagul *<cozi>*, să fie între 1 şi 3 taguri *<coada>*.

- În conținutul unui tag *<coada>*, să existe un atribut *strategie* cu una dintre valorile:
    - *First Come First Served (FCFS)*
    - *Shortest Job First (SJF)*
    - *Priorități*
    - *Round Robin (RR)*
    - *Round Robin (RR) cu priorități*
- Dacă atributul *strategie* are una dintre ultimele două valori, să existe în conținutul aceluiaşi tag *<coada>* atributul *cuanta* cu valori numerice întregi între *1* şi *5*.
- Dacă atributul *strategie* menționat mai sus denotă o strategie cu posibilitatea de preempție forțată şi se vrea acest lucru de către creatorul fişierului, să existe un atribut *preemptiefortata* cu valoarea *da* în conținutul aceluiaşi tag *<coada>*.
- Dacă atributul *strategie* denotă o strategie cu posibilitatea priorităților dinamice şi se vrea acest lucru de către creatorul fişierului, să existe un atribut *prioritatidinamice* cu valoarea *da* în conținutul *<coada>*.
- În tagul *<procesoare>*, să existe ori un tag *<procesor>*, ori un număr egal cu numărul de taguri *<coada>*, de taguri *<procesor>*.
- Dacă strategia cozii procesorului necesită cuantă, în conțintul tagului *<procesor>* mai poate fi adăugat un atribut *cuanta*, care va înlocui atributul *cuanta* din tagul *<coada>* cu indicele egal cu indicele tagului *<procesor>* curent sau 0, dacă este o singură coadă. Acest atribut din conținutul tagului *<procesor>* poate lua valori întregi între *1* şi *5*.
- În tagul *<procese>*, să se găsească între *2* şi *9* taguri *<proces>*.
- În conținutul fiecărui tag *<proces>*, să se găsească un atribut *momentintrare* cu o valoare numerică între *0* şi *9*.
- În conținutul fiecărui tag *<proces>*, să se găsească un atribut *prioritateimplicită* cu o valoare numerică între *0* şi *9*.
- În conținutul fiecărui tag *<proces>*, să se găsească un atribut *prioritatedinamică* cu o valoare numerică între *0* şi *9*.
- În fiecare tag *<proces>* să se găsească un număr impar între *1* şi *9* de taguri *<perioada>*.
- În conținutul fiecărui tag *<perioada>*, să se găsească un atribut *timp* cu o valoare numerică întreagă între *1* şi *5*.

Generate automat

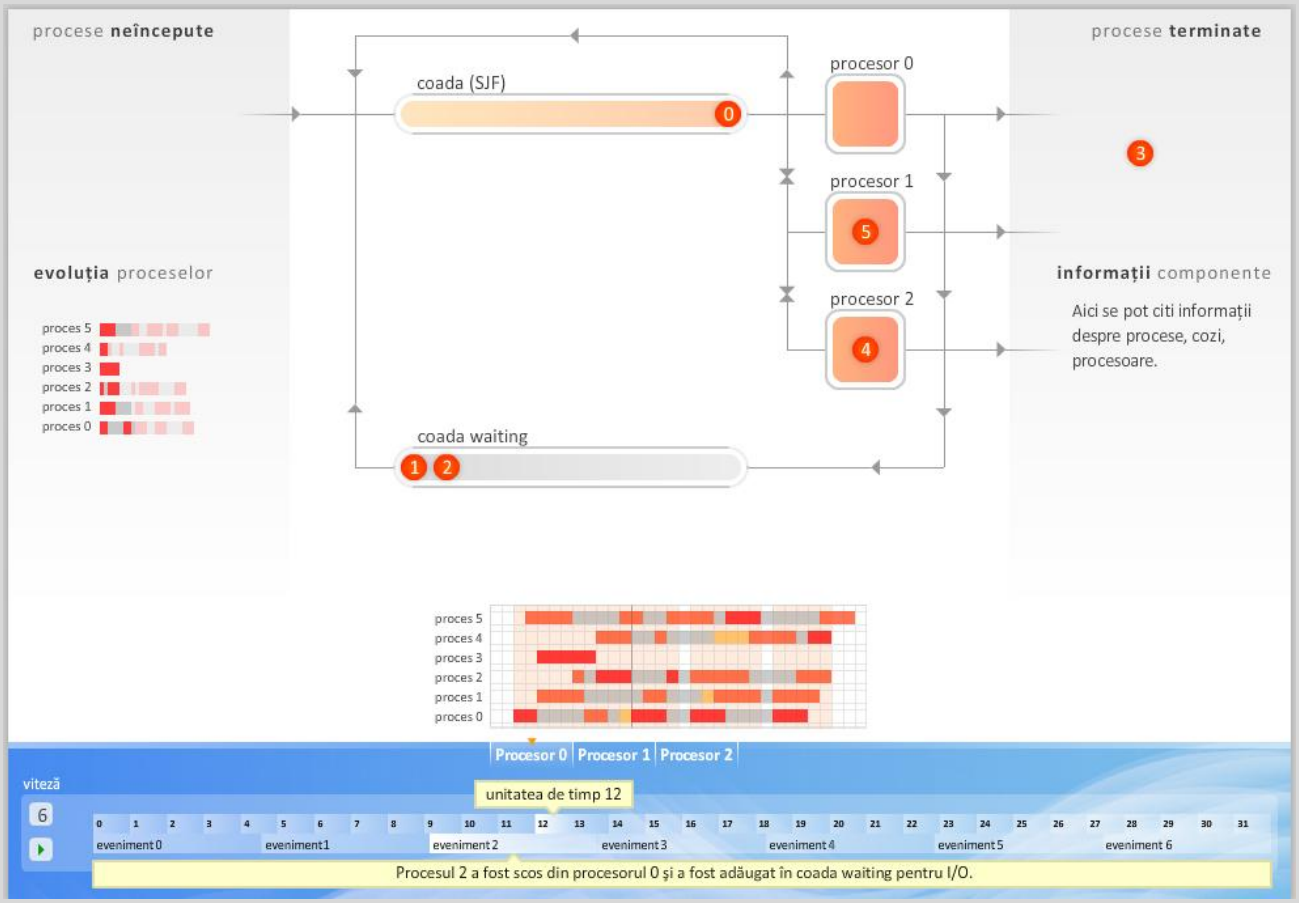
Datele de intrare pot fi generate automat în întregime prin selectarea acestei opțiuni. În acest caz, utilizatorul ajunge direct la etapa de simulare.

Simulare

După introducerea datelor de intrare prin oricare dintre cele 3 metode de mai sus, utilizatorul poate urmări planificarea proceselor.

Interfața simulatorului

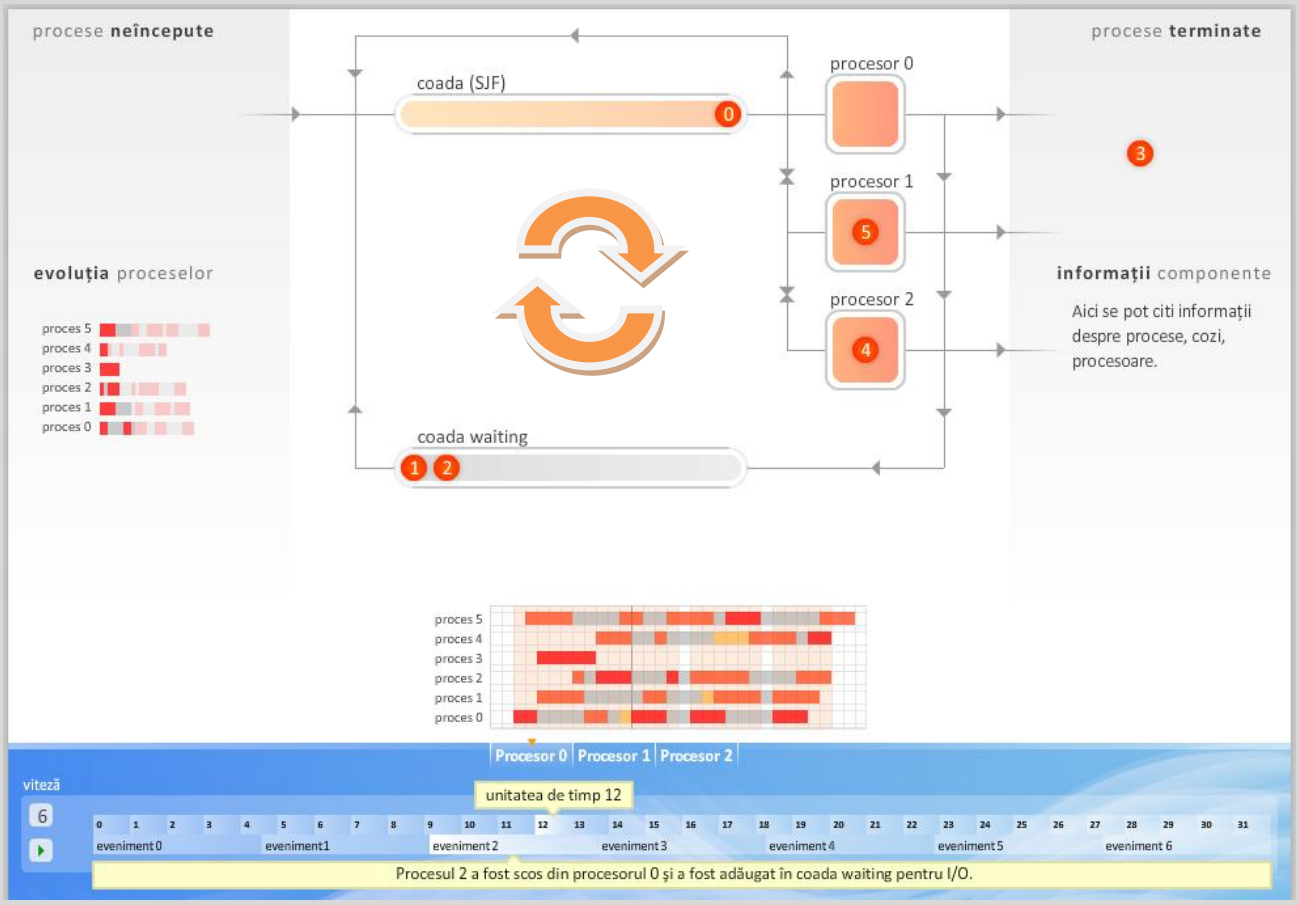
Interfața simulatorului propune urmărirea planificării proceselor într-un flux ce facilitează învățarea intuitivă și în același timp eficientă.



Partea superioară

În partea stângă superioară, se găsesc inițial procesele neincepute, urmând ca la final procesele – terminate – să fie în partea dreaptă superioară.

Partea central superioară ilustrează circular planificarea proceselor. Astfel, trecerea dintr-o coadă ready într-un procesor, realizată de la stânga la dreapta, este completată de trecerea procesor – coadă waiting – coadă ready dinspre dreapta spre stânga.



Partea mediană

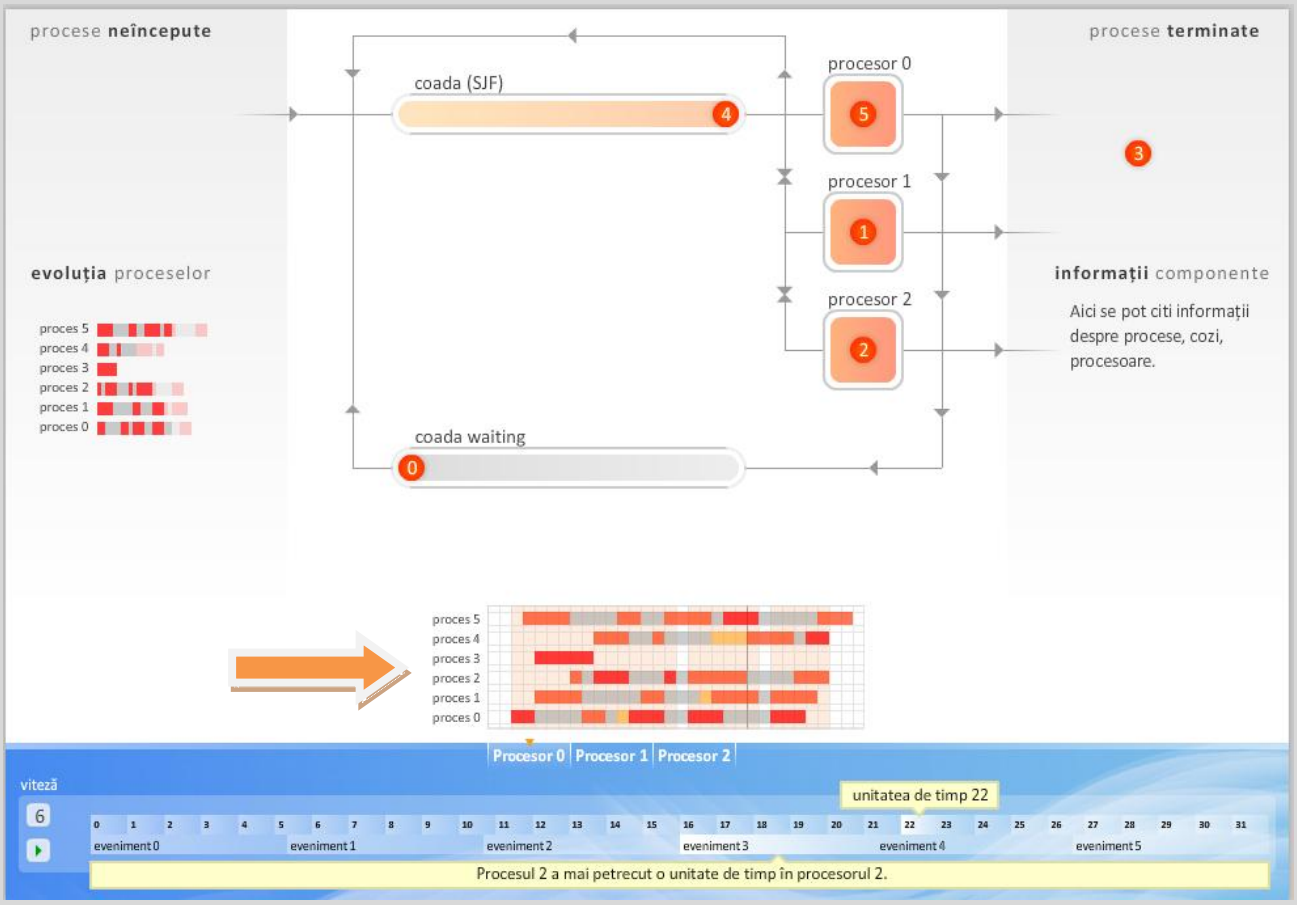
Partea mediană oferă informații mai detaliate despre procese.
În stânga, se poate observa progresul înregistrat de procese la un anume moment de timp.

În centru sunt reprezentate procesoarele cu procesele din cozile lor sau în starea running:

- cu roșu: procesele care rulează în procesorul curent;
- cu portocaliu: procesele care rulează în alte procesoare;
- cu galben: procesele aflate în coada ready a procesorului respectiv;
- cu gri: cele aflate în coada waiting.

Pentru a arăta gradul de utilizare al unui procesor, am marcat cu o nuanță deschisă de portocaliu unitățile de timp când procesorul rulează un proces.

În dreapta, se găsește o secțiune care oferă informații suplimentare despre procesoare, cozi, procese și stările lor în momentul în care cursorul se află pe acestea.

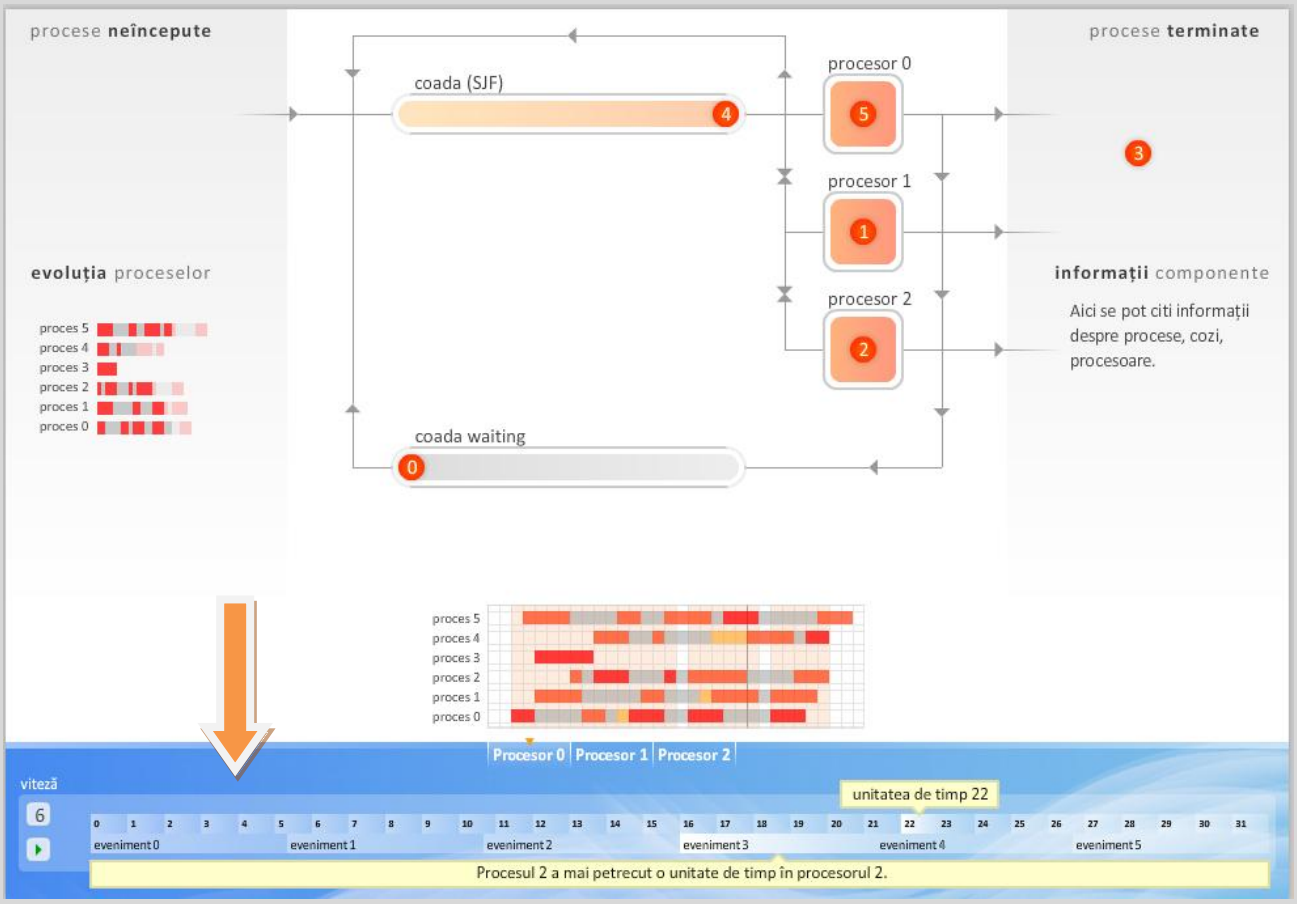


Partea inferioară

În partea inferioară utilizatorul poate vizualiza evoluția proceselor, cozilor și procesoarelor în timp. El poate alege să urmărească o animație cu planificarea proceselor prin specificarea unei viteze (1 – lent, 9 – rapid, 6 –

implicit) și apăsarea pe butonul **>**. La apăsarea lui, butonul se transformă în semnul •, cu semnificația pauză. Animația va fi oprită şi va putea fi reluată din punctul respectiv prin apăsarea aceluiaşi buton.

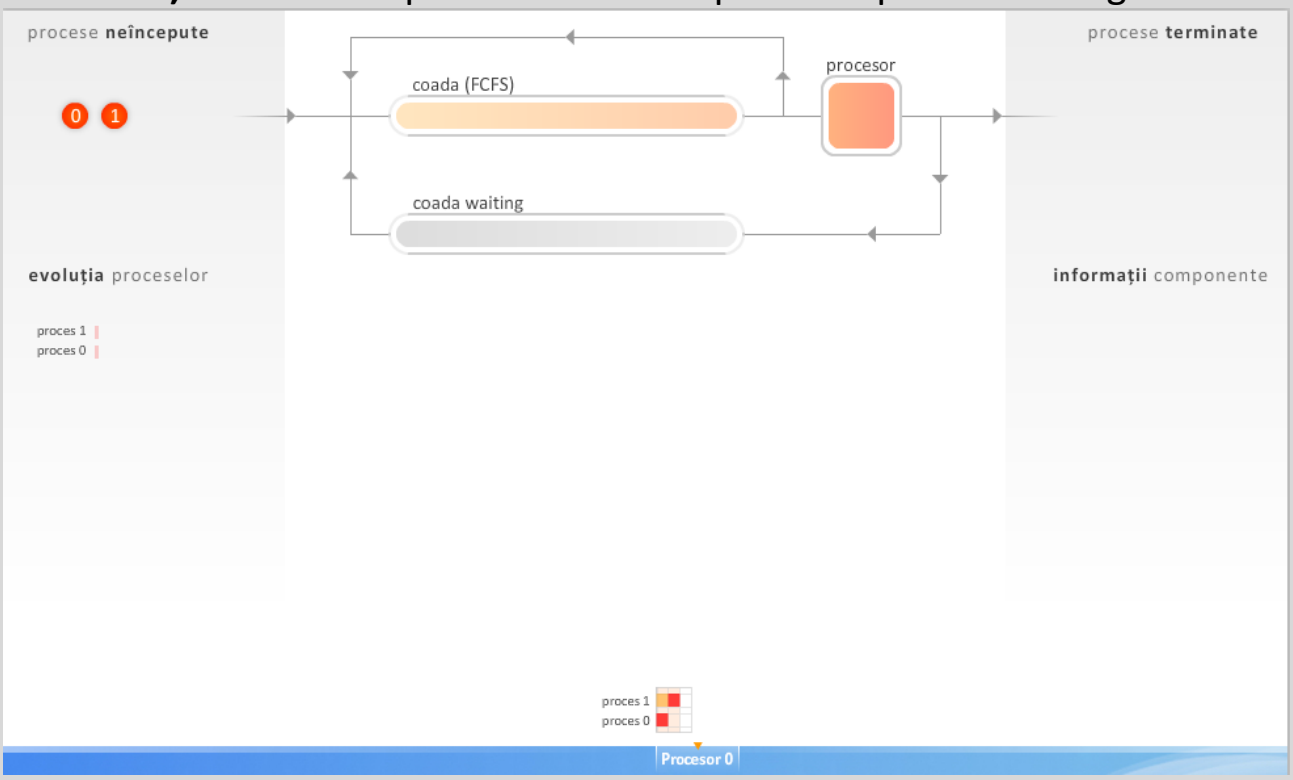
Partea centrală ilustrează unitățile de timp necesare planificării tuturor proceselor pe o bandă numerotată, iar sub aceasta în funcție de unitatea de timp selectată, evenimentele ce au loc în acea unitate de timp (de exemplu: intrarea sau ieșirea unui proces dintr-o coadă, actualizarea priorității dinamice a unui proces, trecerea unui procesor, terminarea unui proces), explicate.



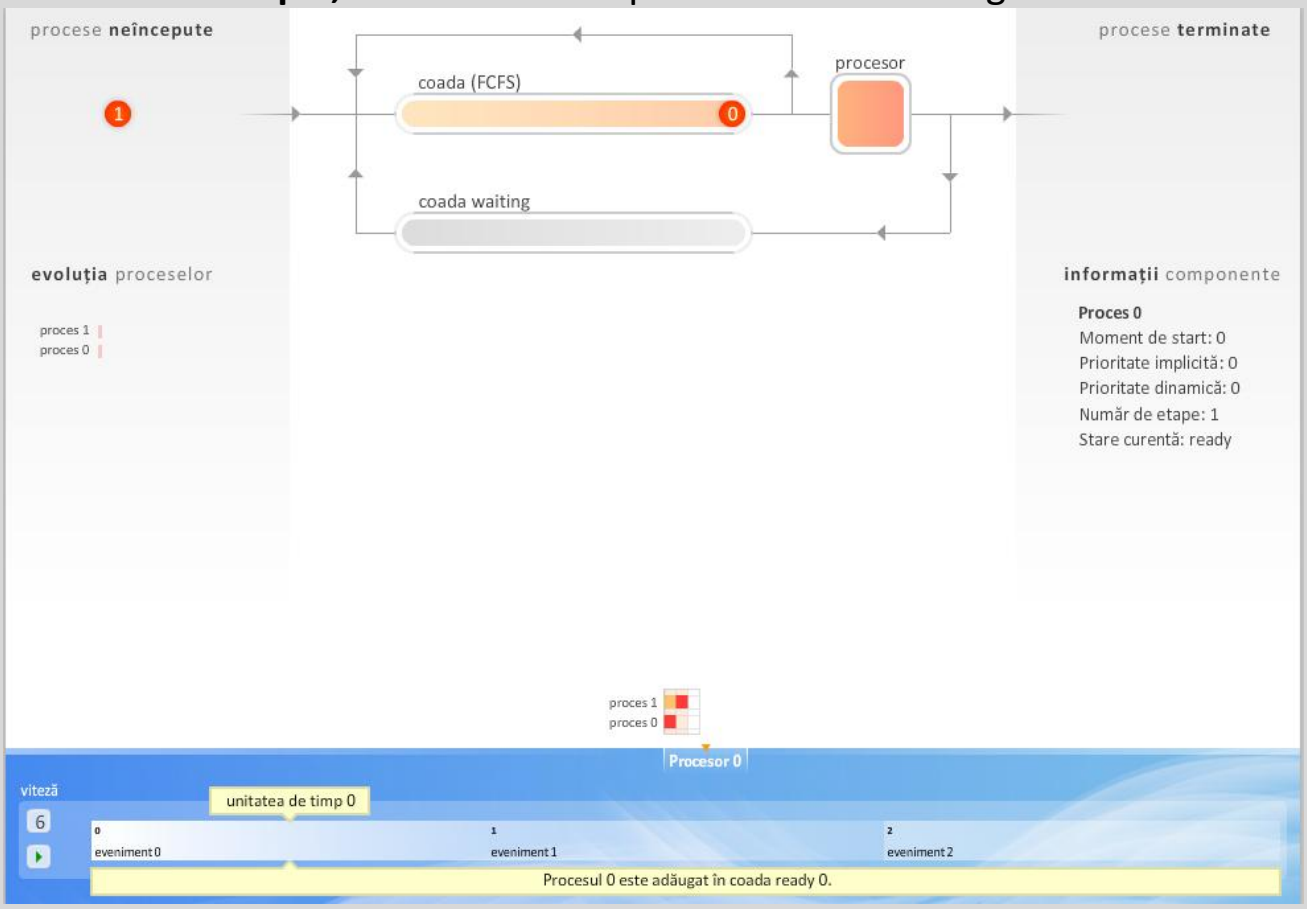
Secvențe demonstrative

Vom ilustra în paginile ce urmează simularea planificării proceselor, considerând că toți parametrii au valori implicite, sistemul este uni-procesor, iar strategia cozii este FCFS.

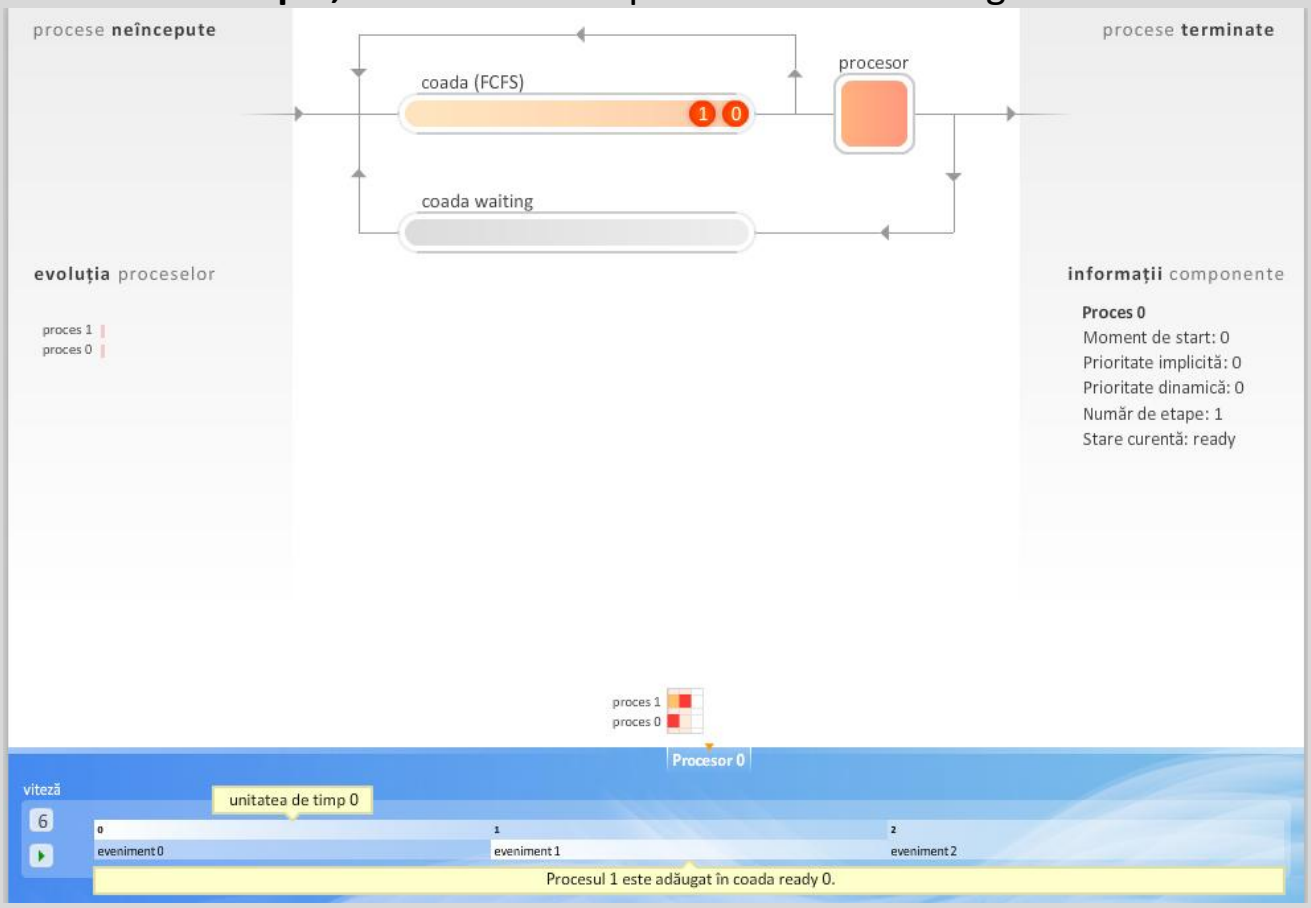
Starea inițială: ambele procese se află în partea superioară stângă.



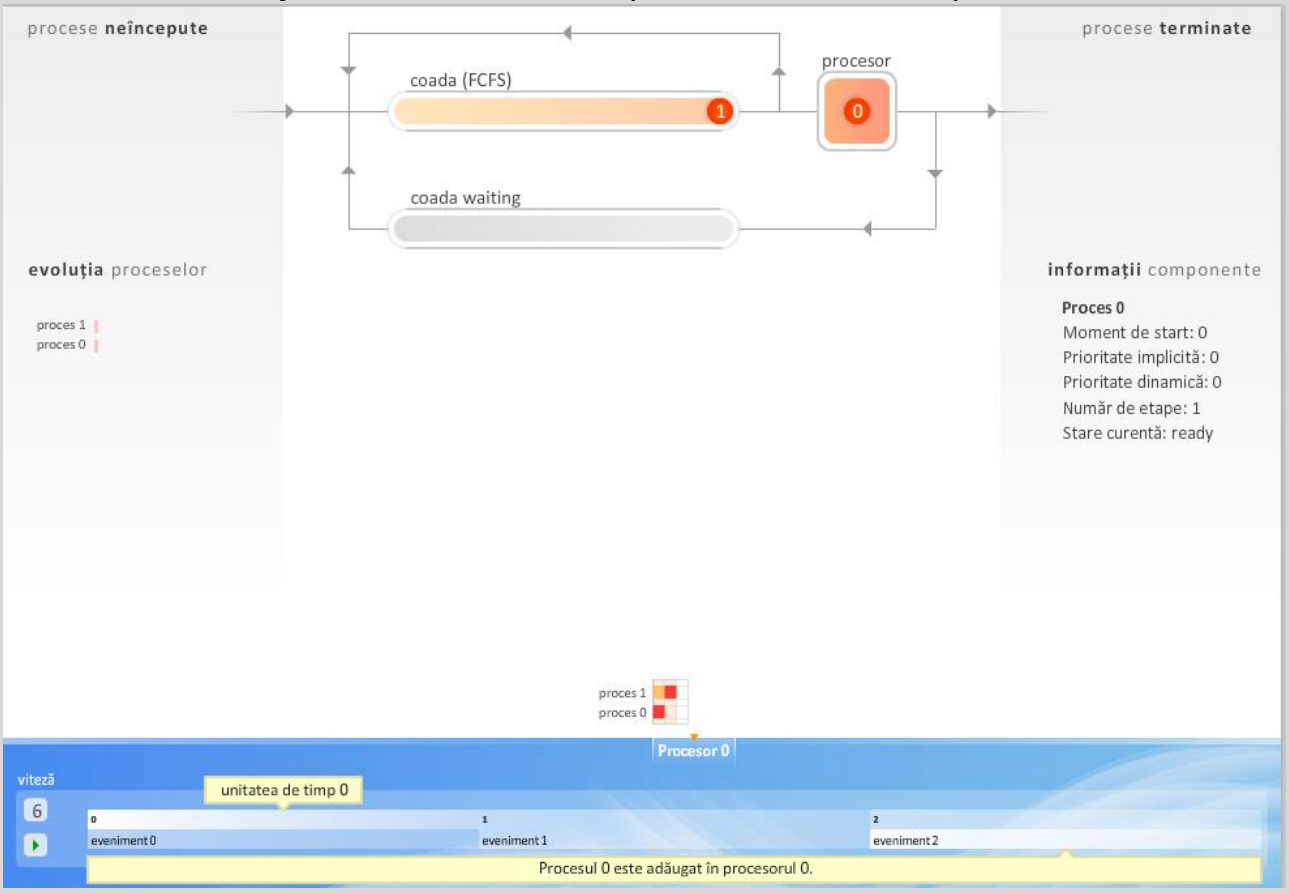
Unitatea de timp 0, evenimentul 0: procesul 0 este adăugat în coada 0.



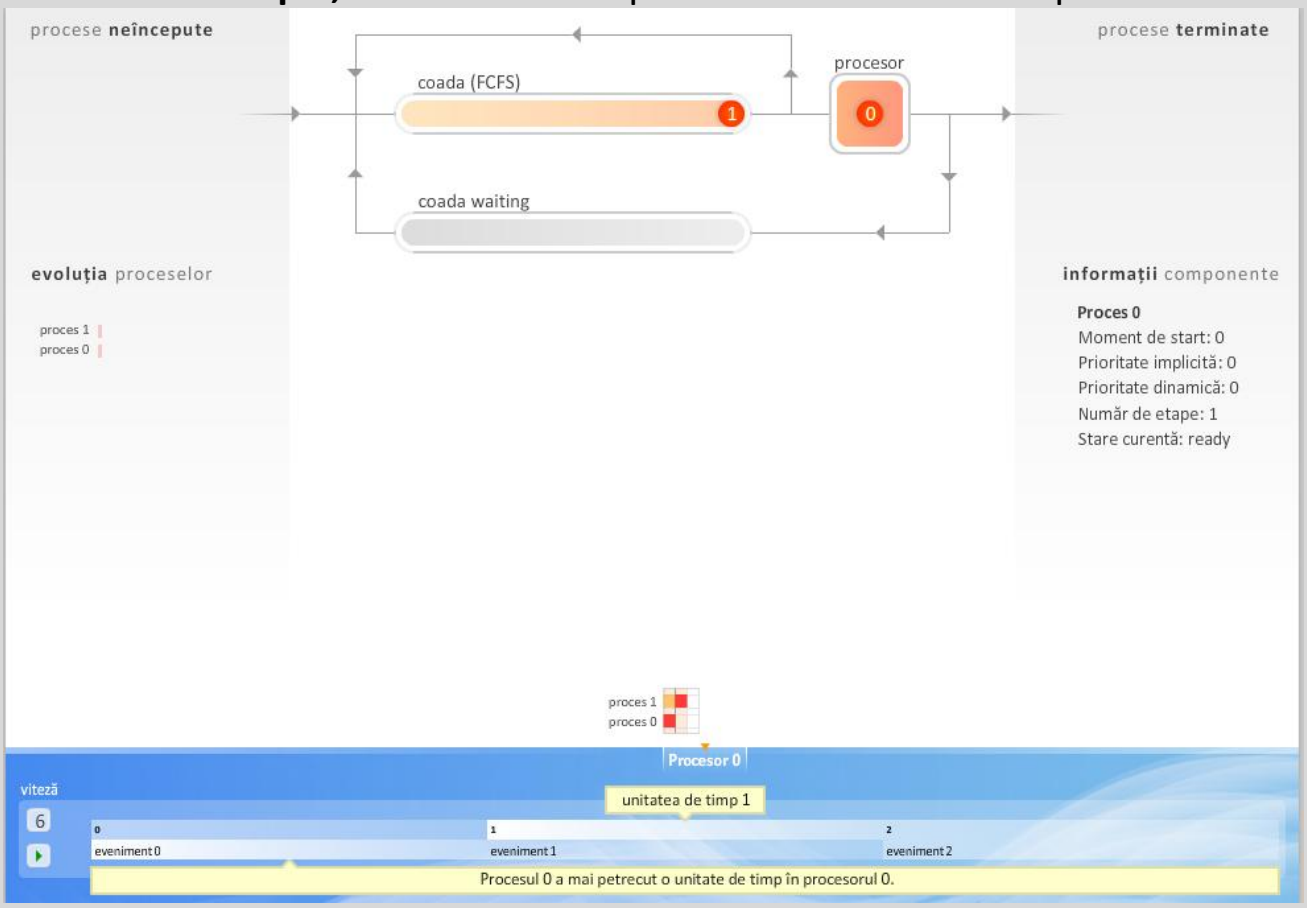
Unitatea de timp 0, evenimentul 1: procesul 1 este adăugat în coada 0.



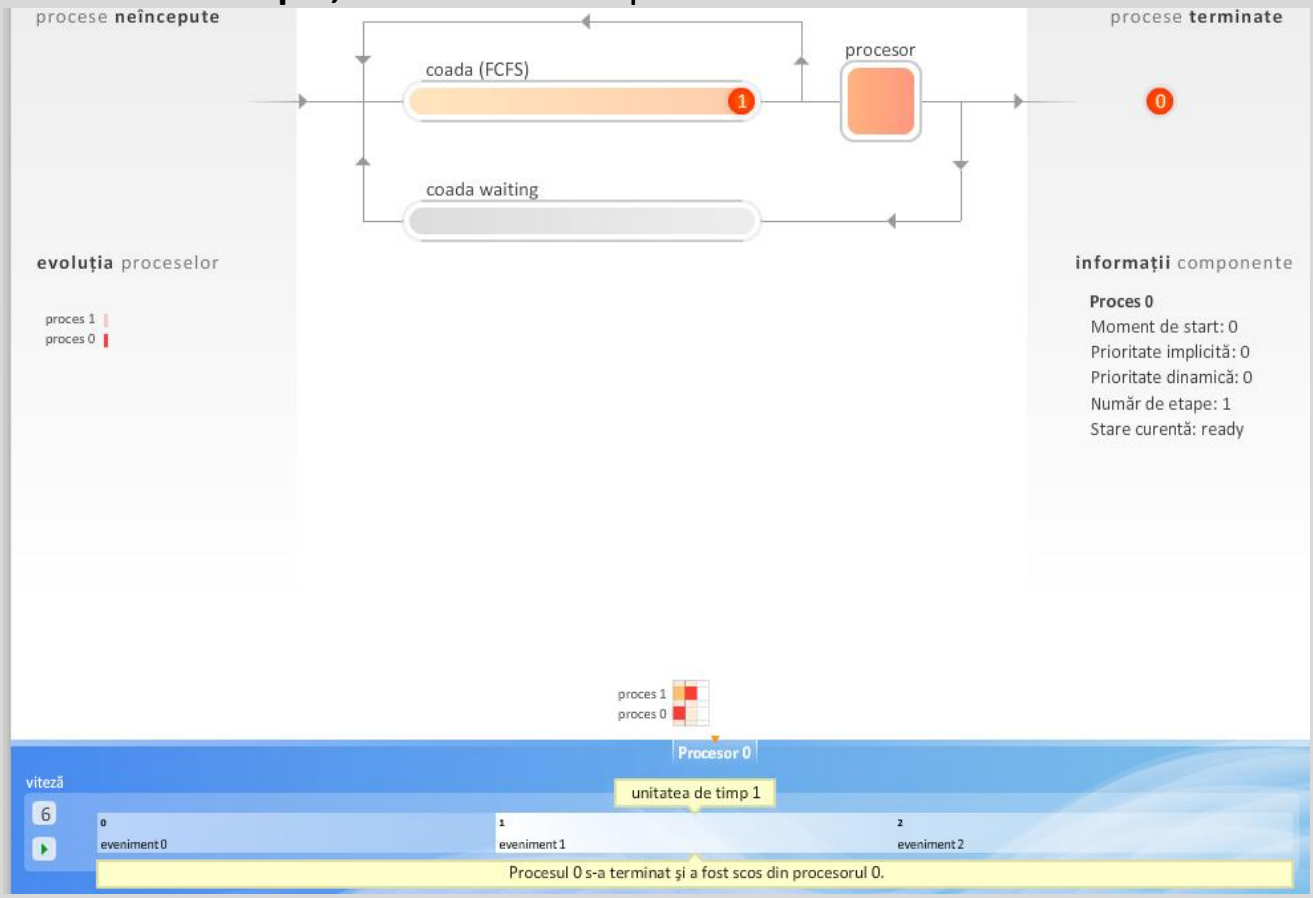
Unitatea de timp 0, evenimentul 2: procesul 0 intră în procesorul 0.



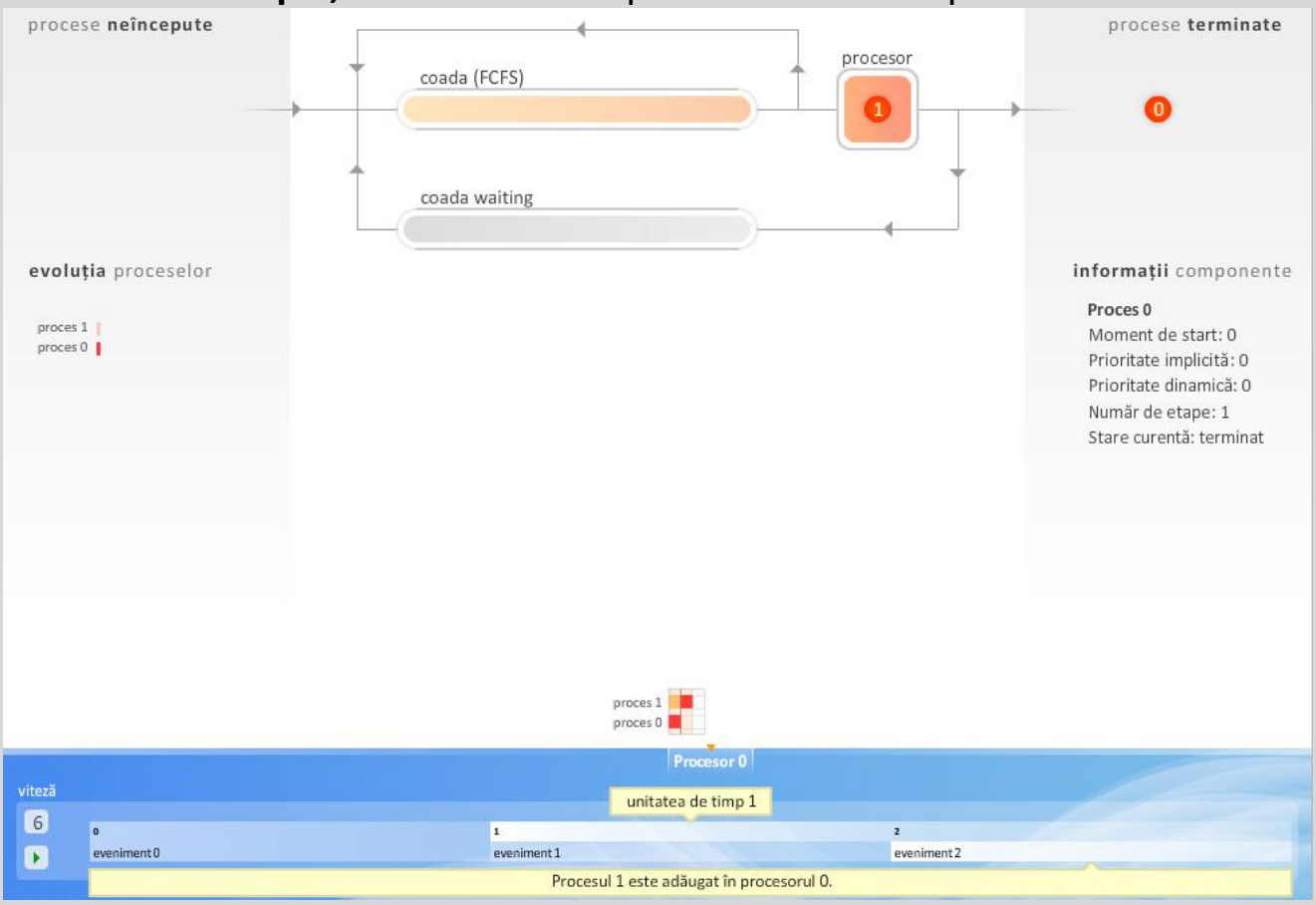
Unitatea de timp 1, evenimentul 0: procesul 0 a mai rulat în procesorul 0.



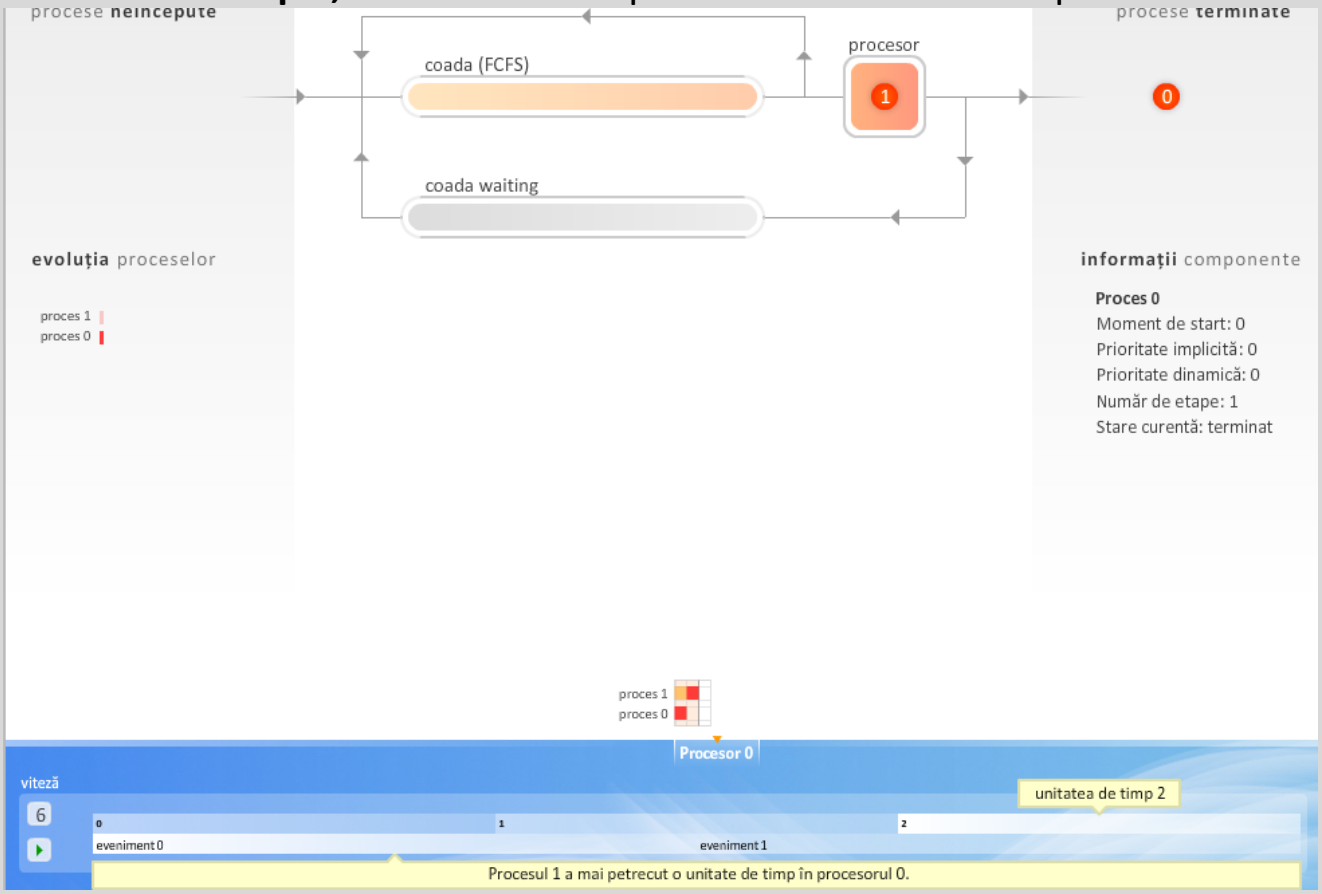
Unitatea de timp 1, evenimentul 1: procesul 0 s-a terminat.



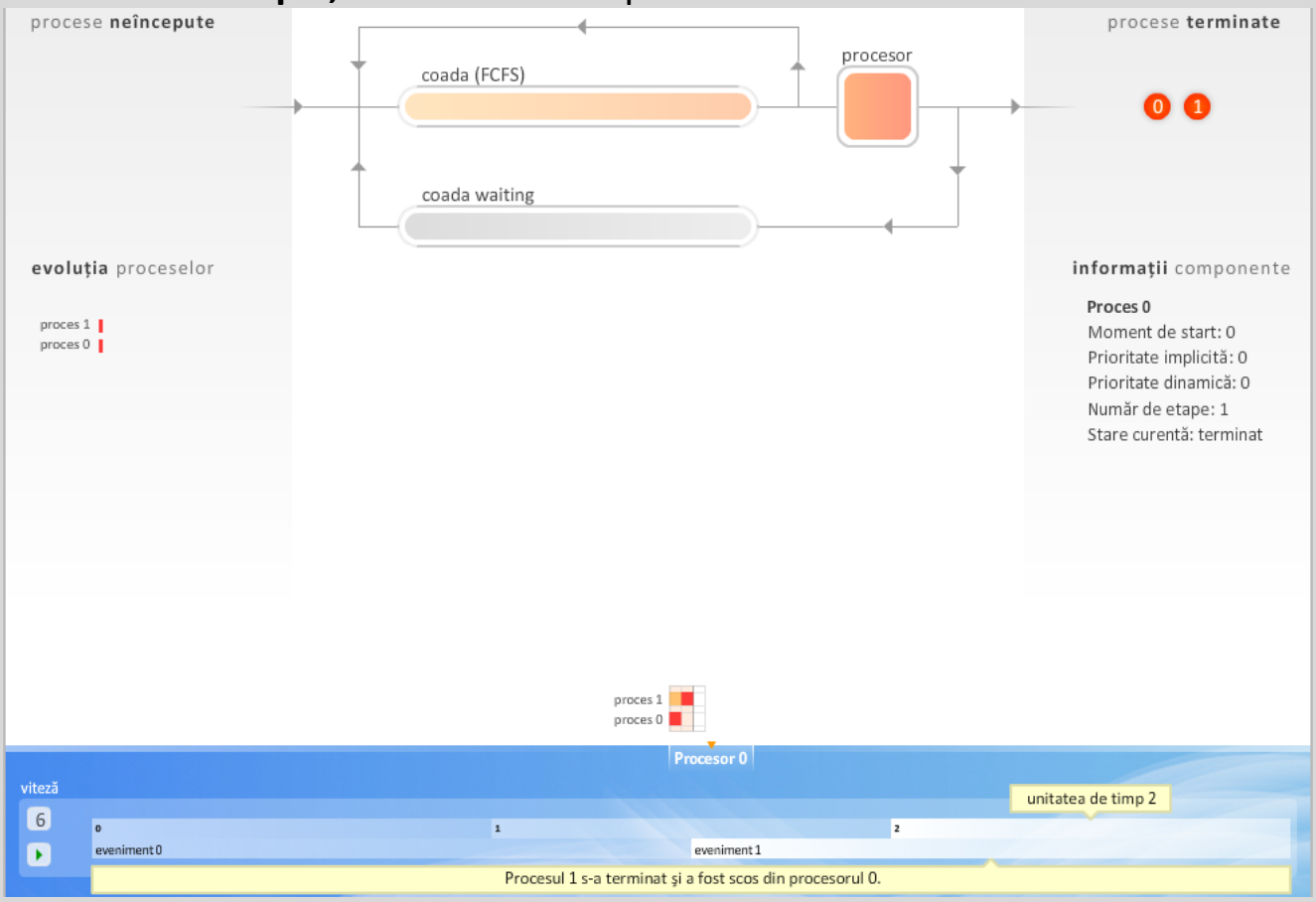
Unitatea de timp 1, evenimentul 2: procesul 1 intră în procesorul 0.



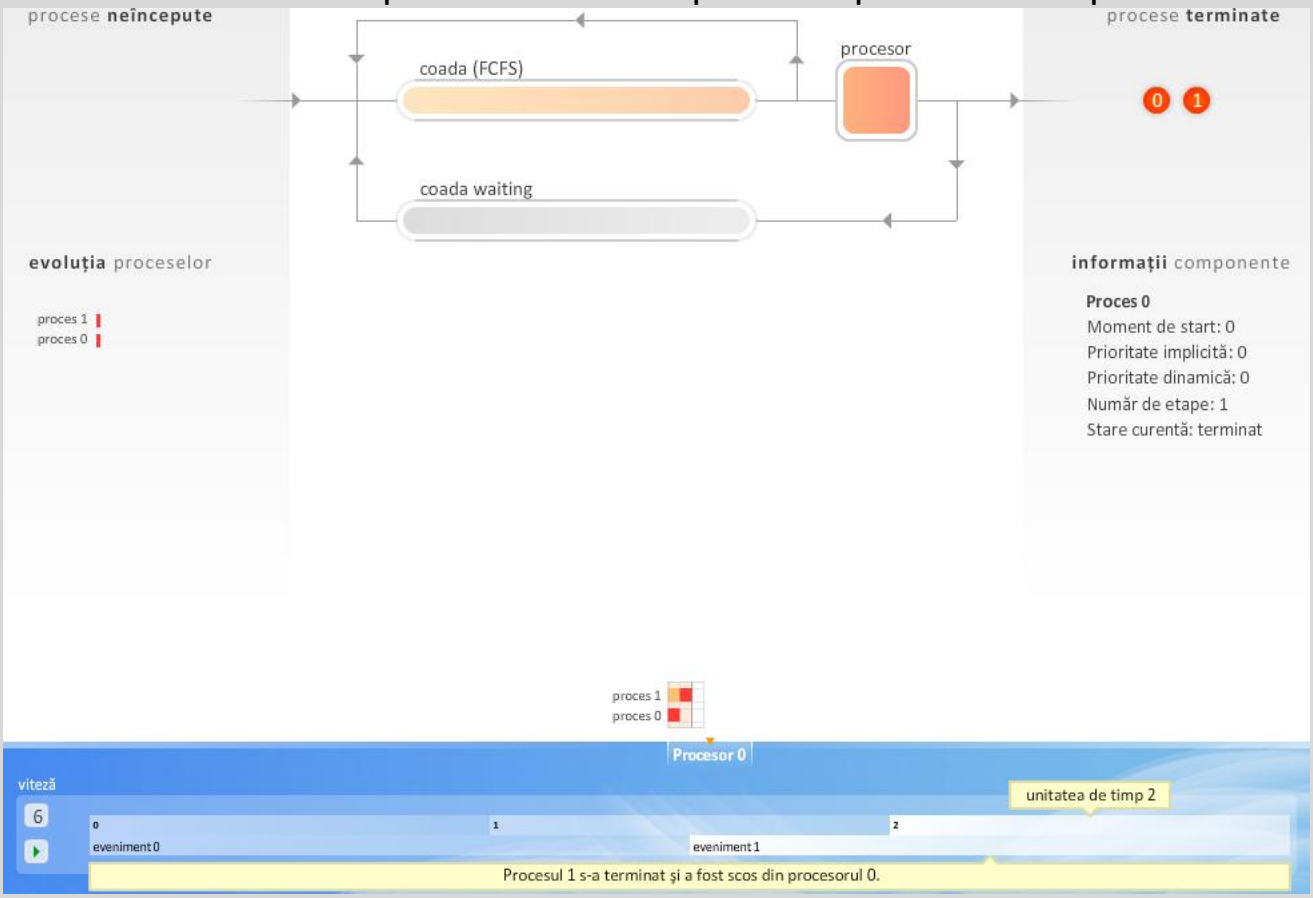
Unitatea de timp 2, evenimentul 0: procesul 1 a mai rulat în procesorul 0.



Unitatea de timp 2, evenimentul 1: procesul 1 s-a terminat.



Starea finală: ambele procese se află în partea superioară dreaptă.



Implementare

Am implementat algoritmii de planificare şi întreaga aplicație folosind Adobe Flash CS3 şi scriind codul în Actionscript2. Înainte de a proiecta aplicația, ne-am documentat din surse pe care le recomandăm tuturor celor care sunt interesați de tematica proiectului nostru.

Documentare

Corectitudinea ştiințifică a proiectului a devenit posibilă în urma documentării din următoarele surse:

- *Operating System Concepts*
  Abraham Silberchatz
  Peter Baer Galvin
  Greg Gagne

- *Curs Sisteme de Operare*
  Cristian Traian Vidraşcu

- *Wikibooks Operating System Design*
  Wikipedia.com

Detalii de implementare

În funcție de datele introduse până la un moment dat, se decide ce date de intrare sunt necesare în continuare. Spre exemplu, dacă utilizatorul alege strategia **Round Robin (RR) cu priorități** pentru o anumită coadă a unui procesor, atunci se vor afişa opțiunile **Preempție forțată** şi **Priorități dinamice**.

Odată ce datele de intrare au fost introduse, se planifică procesele prin procedura de precalcul care urmează:

Cât timp mai sunt procese neterminate:
{
Să presupunem că suntem în unitatea de timp i (incrementată la fiecare iterație).
Pentru fiecare coadă ready cu priorități dinamice şi cu procese în ea:
Actualizează prioritatea proceselor aflate în acea coadă.
Pentru fiecare proces în coada waiting care a terminat instrucțiunile de I/O în unitatea de timp curentă i:
Mută procesul din coada waiting în coada sa ready.
Pentru fiecare proces pentru care timpul de start este egal cu i:
Introduce/Mută procesul în coada cu cele mai puține procese.
Cât timp există două cozi ready astfel încât diferența dintre numărul de procese în prima şi numărul de procese din a doua depăşeşte 1:
Mută procesul din coada mai ocupată în cea mai liberă.
Pentru fiecare proces ce funcționează în interiorul unui procesor:
{
Actualizează timpul petrecut în procesor.
Dacă procesul a terminat instrucțiunile în procesor şi urmează o secvență de instrucțiuni I/O:
Mută procesul din procesor în coada waiting.
Altfel, dacă procesul s-a terminat:
Mută procesul din procesor în procese terminate.
Altfel, dacă politica pentru coada procesorului este RR şi procesul a consumat cuanta procesor:
Mută procesul din procesor în coada ready.
Altfel, dacă politica este cu preempție forțată şi există un proces mai bun în coada procesorului:
Mută procesul din procesor în coada ready.
}
Pentru fiecare procesor liber astfel încât există măcar un proces în coada asociată:
Mută procesul din coadă în procesor.
}

După executarea acestei proceduri, se afişează elementele vizuale pentru utilizator. În acel moment, pentru fiecare unitate de timp, sunt precalculate deja stările şi valorile parametrilor elementelor sistemului.

pentru

FII Competition