Capitolul 3

Simularea Dinamicii Sistemelor Cibernetice din Economie

Simularea constituie, alături de modelare, o metodă de bază în analiza şi proiectarea sistemelor cibernetice din economie. Cu ajutorul acesteia pot fi studiate şi descoperite noi proprietăţi ale sistemelor adaptive complexe. Pe măsură ce s-a dezvoltat modelarea, a evoluat şi simularea, amândouă fiind impulsionată de evoluţia şi creşterea performanţelor calculatoarelor electronice. Frecvent, modelarea şi simularea sunt folosite simultan, în acest mod putând fi cunoscute proprietăţi ale sistemelor care altfel ar fi extrem de dificil de evidenţiat.

Încă din perioada apariţiei sale ca ştiinţă, cibernetica a fost interesată de simulare ca metodă de studiu a sistemelor abordate. O metodă de simulare acceptată şi dezvoltată în cadrul ciberneticii este Dinamica Sistemelor, introdusă în a doua jumătate a anilor 1950 de către Jay W. Forrester, profesor la Massachusetts Institute of Technology (MIT). Jay Forrester a fost angajat la MIT ca profesor de cercetări operaţionale, un domeniu ştiinţific care susţine adoptarea deciziilor manageriale prin utilizarea metodelor ştiinţifice şi matematice. Conform observaţiilor făcute de el, multe dintre metodele cercetărilor operaţionale nu erau adaptate la rezolvarea anumitor probleme manageriale strategice, aceste metode fiind orientate mai mult către optimizare şi obţinerea unor soluţii analitice. De mai multe ori, însă, decidenţii de la nivelele strategice erau interesaţi de tendinţe şi de dinamicile proceselor care aveau loc în sistemele conduse. O altă critică majoră adusă de Forrester metodelor cercetărilor operaţionale era aceea că ele constituiau metode de tip *open-loop* (buclă deschisă), odată ce o soluţie era obţinută, aceasta ne mai putând fi modificată utilizând rezultatele ce se obţin prin aplicarea ei.

Pentru a evita astfel de lucruri, Forrester propune o metodă în buclă închisă (*closed-loop*) în care deciziile sunt privite ca mijloace de influenţare efectivă a mediului înconjurător, schimbările produse în mediu fiind utilizate pentru a îmbunătăţi deciziile la intrarea sistemului astfel încât acestea să conducă la o îmbunătăţire generală la nivelul întregului sistem. Forrester recunoaşte că a ajuns la această concepţie aplicând conceptul de sistem feedback dezvoltat de întemeietorii ciberneticii.

Treptat, dinamica sistemelor s-a dezvoltat, fiind disociată de cercetările operaţionale şi evoluând mult timp ca o disciplină independentă, cu aplicaţii din ce în ce mai ample în tehnică, economie, previziune, teoria deciziilor, management, ştiinţe sociale etc.

Concomitent, prin dezvoltarea calculatoarelor electronice, au apărut şi limbaje de modelare, primul dintre acestea fiind DYNAMO, un limbaj care a făcut posibilă simularea pe calculator a modelelor de dinamică de sistem. Ulterior au fost dezvoltate limbaje din ce în ce mai sofisticate, cum ar fi STELLA sau POWERSIM.

Forrester a insistat mult timp că dinamica sistemelor nu este creată doar pentru simulare. Ea constituie un corp de teorii şi metode de abordare a sistemelor feedback informaţionale. Altfel spus, ştiinţele economice şi sociale pot fi studiate mult mai eficient considerând că sistemele din cadrul lor sunt sisteme de control feedback, adică sisteme ale căror decizii influenţează mediul care, la rândul său, afectează deciziile respective.

Metoda dinamicii de sistem a fost expusă pentru prima oară într-o formă închegată în cartea lui Forrester *Industrial Dynamics* apărută în 1961. În 1968 în cartea *Principles of Systems*, Forrester stabileşte o serie de principii privind comportamentul sistemelor feedback informaţionale şi arată modul în care acest comportament poate fi simulat prin dinamica de sistem. În 1969 Dinamica Sistemelor este aplicată la studierea dinamicii urbane (Forrester, 1969). Dar cel mai mare succes l-a avut această metodă atunci când a fost aplicată pentru studierea dinamicii globale (Forrester, 1973) într-un studiu pentru Clubul de la Roma, o asociaţie a oamenilor de ştiinţă şi personalităţilor politice care erau interesaţi de viitorul omenirii. Raportul pentru Clubul de la Roma (Meadows, et al., 1974) a apărut în mai multe ediţii, fiind tradus în 20 de limbi. Deşi concluziile acestui raport sunt astăzi perimate şi chiar discutabile, proiectul în sine reprezintă unul dintre primele în care metode ale ciberneticii şi simulării pe calculator au fost aplicate pe scară largă.

Ulterior, dinamica de sistem a fost aplicată în diferite alte domenii cum ar fi: managementul, finanţele, producţia, desfacerea, studiul pieţelor internaţionale, studierea interacţiunilor om-calculator etc.

Cu toate rezultatele pozitive obţinute, dinamica de sistem este o metodă care depinde de calitatea modelului elaborat, ceea ce face ca, de multe ori, în studierea sistemelor adaptive complexe, să se manifeste rigidităţi şi limite date de apartenenţa la clasa modelelor bazate pe ecuaţii. Apariţia şi dezvoltarea modelelor bazate pe agenţi au permis dezvoltarea unor metode de simulare în care sunt implicaţi agenţi care se comportă independent unul de altul, dar între care se creează interacţiuni şi conexiuni care determină, în ultimă instanţă, un anumit comportament al întregului sistem care nu putea fi anticipat înainte de simulare. Simularea bazată pe agenţi reprezintă un domeniu cu o dezvoltare rapidă atât în

ceea ce priveşte principiile teoretice şi aplicative, cât şi a limbajelor de modelare şi simulare utilizate. Limbaje de modelare cum sunt SWARM, REPAST, JAS, NETLOGO ş.a. sunt astăzi din ce în ce mai frecvent utilizate pentru a simula sisteme bazate pe agenţi sau sisteme multiagent.

În cadrul acestui capitol vom prezenta cele două metode principale de simulare: dinamica de sistem a lui Forrester şi simularea sistemelor bazate pe agenţi. Vom arăta principiile care stau la baza acestor două metode, specificul limbajelor de simulare utilizate de fiecare dintre ele precum şi asemănările şi deosebirile care se înregistrează în aplicarea lor în practică în diferite domenii ale economiei.

3.1. Metoda dinamicii de sistem

Conform lui Jay Forrester, dinamica de sistem este o teorie asupra structurii şi comportamentului sistemelor complexe. Această structură are patru nivele:

- Limite închise;
- Bucla feedback ca o componentă de bază a sistemului;
- Nivele şi ritmuri;
- Scopuri, condiţii observate, diferenţa dintre scopuri şi condiţii şi acţiuni dorite.

O premisă importantă în dinamica de sistem este că comportamentul unui sistem este determinat de caracteristicile întregului şi nu de cele ale părţilor individuale (aceasta este una dintre deosebirile esenţiale faţă de simularea bazată pe agenţi). Cu alte cuvinte, dinamica de sistem porneşte de la ipoteza de bază că sistemul are limite închise. Deşi între sistem şi mediul său înconjurător au loc tot timpul schimburi materiale, acestea nu fac ca limitele sistemului să devină difuze, fiind cunoscut încă de la început între ce limite se situează sistemul respectiv.

Acest lucru se numeşte „închidere cauzală”, deci limitele închise separă interiorul sistemului care este dinamic semnificativ, de mediul acestui sistem care este dinamic nesemnificativ, adică schimbările care au loc în exterior nu sunt menite să schimbe semnificativ ceea ce se petrece în interior. Drept consecinţă, toate relaţiile dintre elementele sistemului care sunt considerate importante pentru a explica comportamentul dinamic vor fi închise în modelul de dinamică de sistem.

În dinamica de sistem comportamentul întregului sistem este determinat de structura buclelor feedback interdependente din interiorul limitelor închise. Deci această metodă tinde să privească sistemul mai degrabă dintr-o perspectivă endogenă decât dintr-una exogenă: sistemul se comportă într-un anumit mod datorită structurii sale interne decât ca rezultat al factorilor exteriori.

Acest lucru implică faptul că în cadrul sistemului deciziile sunt încorporate în bucle feedback. Aceste bucle feedback ştim că pot fi pozitive sau negative. O

buclă pozitivă determină acţiuni care cresc mărimile de stare ale sistemului care, la rândul lor, duc la alte acţiuni ce cresc stările sistemului. Cu alte cuvinte, o buclă feedback pozitivă este cauza unor procese de creştere continuă (evident că se poate extinde şi asupra cazurilor de descreştere continuă). Un exemplu comun în acest sens este spirala salarii-preţuri. Creşterea salariilor conduce la preţuri mai mari ale bunurilor produse care, la rândul lor, determină o nouă creştere a salariilor ş.a.m.d.

O buclă feedback negativă este acea buclă în care o creştere a unei mărimi de stare duce, după o anumită perioadă, la descreşterea aceleiaşi mărimi. Bucla feedback negativă este, prin natura ei, stabilizatoare, comportamentul întregului sistem fiind orientat către atingerea unui anumit scop, în jurul căruia au loc ulterior oscilaţii de amplitudine din ce în ce mai redusă. Un exemplu de buclă feedback negativă din economie este raportul dintre inflaţie şi şomaj, exprimat matematic prin curba Phillips. Astfel, o creştere a ratei anticipate a inflaţiei îi face pe producători să întrevadă posibilitatea de a vinde o cantitate mai mare de produse pe piaţă. Aceştia vor angaja mai mulţi muncitori, deci rata şomajului se va reduce. Creşterea numărului de angajaţi va duce la o producţie mai mare, deci la o ofertă pe piaţă crescută. Acest lucru, însă, face ca preţul bunurilor pe piaţă să înceapă să scadă. Scăderea observată a preţurilor duce la o scădere a ratei anticipate a inflaţiei. De această dată, producătorii vor limita angajările sau chiar vor concedia angajaţi. Drept urmare, rata şomajului va creşte.

Orice buclă feedback dintr-un model de dinamică de sistem conţine cel puţin un nivel şi un ritm. Nivelele (sau stocurile) reprezintă acumulări în sistem, de exemplu cantitatea de produse aflată în stoc sau numărul de muncitori angajaţi la un moment dat de timp. De regulă, în diagramele modelelor de dinamică de sistem, nivelele sunt reprezentate prin pătrate (▭) şi sunt însoţite de intrări şi ieşiri de fluxuri materiale, valorice, informaţionale, de oameni sau de comenzi.

Ritmurile sunt reprezentate în modele cu simbolul valvelor (⧖) şi însoţesc atât la intrare cât şi la ieşire un nivel. Se exprimă astfel ideea că dintr-un nivel intră şi respectiv ies cantităţi care sunt controlate cu ajutorul acestor ritmuri.

Pentru a fi mai expliciţi în privinţa raporturilor dintre nivele şi ritmuri să considerăm o structură a unui model de dinamică de sistem ca cea din figura 3.1. Stocurile (deci nivele) sunt: angajaţi (oameni), contul curent (bani) şi stocul de produse (număr de produse aflate în stoc). Aceste trei mărimi pot fi considerate ca fiind stările sistemului, iar valorile acestora se pot modifica datorită schimbărilor în fluxuri (fluxul de angajaţi, fluxul de încasări şi fluxul de produse).

De exemplu, stocul de produse va creşte ca urmare a intrării produselor realizate şi va descreşte ca urmare a livrării produselor către clienţi. Contul curent va creşte prin încasarea facturilor produselor vândute şi va descreşte ca urmare a cheltuielilor realizate (plată materii prime, a utilităţilor, a salariilor, a dobânzilor la

bancă etc.). În sfârşit, numărul de muncitori va creşte ca urmare a plecării sub diferite motive a unor angajaţi.

Simbolurile sub forma unor nori ( ) reprezintă limitele sistemului, în sensul că prin acele locuri intră în sistem muncitori, bani sau materii prime şi materiale necesare activităţilor de producţie şi, de asemenea, ies din sistem aceleaşi lucruri atunci când acestea sunt trimise către alte sisteme sau clienţi.

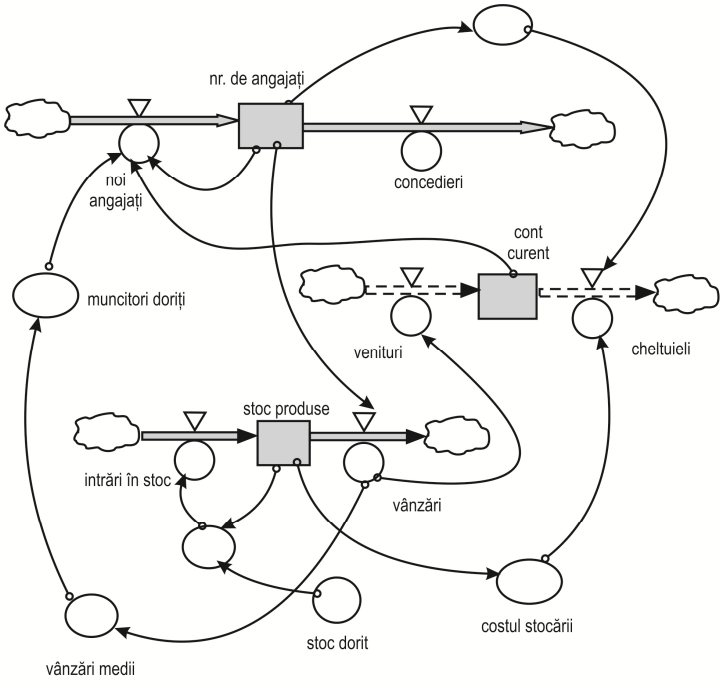


Figura 3.1.

Fluxurile din acest model sunt controlate prin politici decizionale care, la rândul lor, sunt încorporate unor bucle feedback. De exemplu, în cazul stocului de produse, dacă stocul dorit (o variabilă auxiliară care se reprezintă în dinamica de sistem printr-un cerc ( ) depăşeşte nivelul stocului existent (nivel) o decizie va fi luată pentru a achiziţiona mai multe produse şi a creşte astfel stocul existent. Drept urmare, diferenţa dintre stocul dorit şi cel existent se va reduce. Cu cât această diferenţă devine mai mică, cu atât comenzile pentru noi produse finite vor fi mai reduse. Apare astfel o buclă feedback ce are drept scop reducerea şi chiar eliminarea diferenţei stoc dorit – stoc existent. Invers, dacă stocul existent este mai mare decât stocul dorit, comenzile de noi produse vor fi zero până când stocul existent se va reduce şi va deveni egal cu stocul dorit. Acest lucru se face prin vânzări ale produselor către clienţi.

Vânzările vor reduce stocul, dar vor genera şi venituri care vor afecta fluxul de intrare în contul curent. Veniturile din contul curent vor creşte. Cu cât vânzările vor fi mai mari cu atât veniturile vor creşte mai repede. Acest lucru este arătat prin săgeata care leagă vânzările de venituri.

În model se observă că există o a doua săgeată de la vânzări către vânzările medii. Dacă vânzările cresc (sau scad) pe o anumită perioada de timp, atunci vânzările medii vor creşte (scădea). Acest indicator va influenţa numărul de muncitori doriţi care va fi mai mare dacă vânzările medii sunt mari şi, respectiv, mai mic dacă vânzările medii sunt mai mici.

Comparându-se numărul de muncitori doriţi cu angajaţii existenţi se va determina dacă vor avea loc noi angajări. Cu cât diferenţa este mai mare, cu atât vor fi angajaţi mai mulţi muncitori. Dar acest lucru face să crească nivelul muncitorilor existenţi. Numărul acestora determină cheltuielile cu salariile, deci şi cheltuielile totale ale sistemului. Contul curent va fi afectat, ceea ce va duce la reducerea ratei de angajare de noi muncitori.

Se observă că modelul conţine trei bucle feedback interdependente care, prin acţiunea lor, determină un anumit comportament al întregului sistem. Fiecare buclă feedback conţine cel puţin un nivel şi o variabilă de ritm. Buclele feedback introduse încearcă să elimine o diferenţă dintre un nivel dorit şi un nivel existent. Deciziile luate pentru a realiza acest lucru sunt aplicate ritmurilor de intrare sau de ieşire din nivele. În urma acestor decizii, variabilele de nivel (stocurile) ale sistemului se modifică în direcţia nivelelor dorite. Între bucle feedback există influenţe reciproce care pot determina accelerarea sau întârzierea convergenţei nivelelor existente către nivelele dorite.

Simularea cu ajutorul modelelor de dinamică de sistem încearcă să stabilească cele mai bune decizii care urmează sa fie adoptate astfel încât stările dorite sa fie atinse. Marea problemă care apare este aceea că combinaţiile de bucle feedback generează relaţii neliniare între variabilele de stare. Aceste relaţii neliniare conduc, de multe ori, la comportamente extrem de complicate care pot fi observate abia după ce se realizează o simulare pe termen îndelungat. Rezultatele care se obţin, în acest caz, nu mai sunt utile pentru conducerea sistemului respectiv care se poate confrunta cu schimbări dramatice de comportament ale întregului sistem sau ale unor părţi ale acestuia.

Unele dintre buclele feedback din structura modelelor pot prezenta întârzieri, adică perioade destul de lungi după care efectele deciziilor care afectează ritmurile sunt vizibile. Diferenţele dintre întârzierile caracteristice buclelor feedback precum şi neliniarităţile care se manifestă pot produce comportamente neaşteptate sau contraintuitive.

Totuşi, în ansamblul ei, metoda dinamicii de sistem se dovedeşte foarte utilă în cunoaşterea şi explicitarea comportamentului dinamic ale sistemelor feedback din economie.

3.2. Procesul de construire al unui model de dinamică de sistem

Construirea unui model de dinamică de sistem presupune parcurgerea mai multor etape, şi anume:

- Identificarea problemei şi scopului modelului;
- Conceptualizarea sistemului;
- Formularea modelului şi estimarea parametrilor;
- Analiza comportamentului modelului: testarea şi analiza de senzitivitate;
- Evaluarea modelului: validarea modelului;
- Analiza politicilor decizionale;
- Utilizarea modelului sau implementarea.

Aceste etape arată destul de clar scopul principal al dinamicii de sistem, şi anume de a construi un model al sistemului cu care pot fi realizate experimente având ca obiective:

1) O mai bună înţelegere a structurii şi comportamentului sistemului;
2) Proiectarea de politici robuste pentru rezolvarea problemelor care apar în sistem.

Avantajul principal al construirii unui model de simulare bazat pe dinamica de sistem constă, de fapt, în posibilitatea de a realiza un număr mare de experimente, lucru care nu ar fi posibil în realitate. Procesul de construire a modelului de dinamică de sistem, după cum se observă, începe cu identificarea unei probleme şi sfârşeşte cu implementarea unei soluţii. Dar acest proces nu este linear ci este iterativ sau ciclic. Există multe etape care se reiau până când o politică decizională potrivită este identificată. Acest caracter ciclic reprezintă şi un indicator al faptului că procesul de construire a modelului cuprinde o mare cantitate de învăţare. Să examinăm în continuare mai în detaliu fiecare etapă.

3.2.1. Identificarea problemei şi a scopului modelului

Pentru a construi un model este necesar să avem definit un scop clar şi să ne concentrăm mai mult asupra problemei şi nu a sistemului. Acest lucru pare paradoxal având în vedere faptul că dinamica de sistem se orientează către sisteme. Totuşi, este evident faptul că existența unui scop clar definit va face ca întregul proces de elaborare a modelului să fie mai eficient.

Se poate întâmpla ca în această etapă să se identifice mai multe scopuri. În forma sa cea mai simplă, scopul presupune o înţelegere mai bună a efectelor potenţiale ale unor evoluţii curente observate în sistem.

Etapa începe cu identificarea unui comportament problematic al sistemului, cum ar fi, de exemplu: descreşterea satisfacţiei clienţilor, scăderea vânzărilor, creşterea şomajului etc. Aceste evoluţii problematice pot fi reprezentate în timp prin grafice care dau „modul referenţial al comportamentului” şi care vor fi utilizate pentru a compara ulterior efectele unor politici decizionale. Acest concept

al „modului referenţial de comportament" este central în întregul proces de construire a modelului. Unul sau mai multe astfel de reprezentări grafice vor exprima, în esenţă, care sunt problemele pe care urmează să le abordeze şi să le rezolve modelul de dinamică de sistem. La sfârşitul acestui proces, prin utilizarea modelului respectiv vor fi generate aceleaşi grafice dar cu îmbunătăţirile sugerate de model în politicile decizionale ce vor fi adoptate, reprezentând o modalitate de creştere încrederii în model.

Se pot, totuşi, întâlni situaţii în care aceste moduri referenţiale de comportament nu pot fi stabilite datorită absenţei datelor necesare. În economie acest lucru se întâlneşte destul de frecvent, mai ales în cazul unor sisteme în care nu au fost încă realizate sau a căror activitate este prea recentă pentru a putea avea date suficiente pentru a reprezenta comportamentul problematic. În acest caz, totuşi, procesul de construire a modelului ca şi politicile decizionale generate de acesta pot fi utilizate ca un mijloc de studiere a dinamicii potenţiale a structurii identificate în model.

Un element important al identificării problemei este şi orizontul de timp. Este clar că acesta afectează modul în care problema urmează să fie studiată. De exemplu, dacă analiză ecuaţia costurilor unui program de sănătate, acest lucru se va face diferit dacă orizontul de timp este de 3 ani sau de 15 ani.

3.2.2. Conceptualizarea sistemului

Un concept important al dinamicii de sistem este cel de limite ale sistemului. Limitele determină ceea ce se consideră că aparţine sistemului şi ceea ce nu aparţine acestuia, ci mediului său extern. Dacă s-a identificat problema şi scopul urmărite de model este uşor să se stabilească ce elemente pot fi omise fără să se afecteze studierea sistemului sau să se reprezinte greşit comportamentul problematic al acestuia.

După ce au fost definite limitele sistemului, conceptualizarea sistemului poate începe. Conceptualizarea este în general realizată într-un mod vizual. Există două tipuri de diagrame ce sunt utilizate: diagrama buclelor cauzale şi diagrama de flux. În figura 3.1 a fost reprezentată o diagramă de flux. Diagramele buclelor cauzale constituie un mijloc puternic de a exprima sintetic relaţiile cauzale din sistem şi de a identifica procesele feedback. Într-o astfel de diagrama, a buclelor cauzale, o relaţie cauzală între două variabile se reprezintă cu ajutorul unei săgeţi orientate pe care se indică prin semnul + sau -, tipul de relaţie. O relaţie cauzală pozitivă implică faptul că ambele variabile se vor schimba în aceiaşi direcţie. De exemplu:

(+)

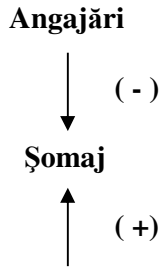
Numărul de automobile → Poluarea aerului

O relaţie cauzală negativă implică faptul că variabilele se vor schimba în direcţii diferite. De exemplu:

(-)

Numărul de automobile ⟶ Numărul de călători cu trenul

Deci semnele asociate relaţiilor cauzale depind doar de relaţia dintre cele două variabile şi nu de creşterea sau descreşterea variabilelor. De regulă, în modelele de dinamică de sistem putem întâlni cazuri în care o variabilă este afectată cauzal de două sau mai multe variabile. De exemplu:



Se observă, astfel, că numărul de angajări scade şomajul, în timp ce oferta de muncă de pe piaţa forţei de muncă (deci numărul de oameni care îşi caută slujbe) duce la creşterea şomajului. Într-un model de dinamică de sistem relaţiile cauzale se pot extinde, cuprinzând din ce în ce mai multe variabile. Unele dintre relaţii pot conduce la o formă de bucle feedback, deci bucle în care se formează bucle închise care cuprind două sau mai multe variabile. Una dintre cele mai simple bucle feedback este cea a raportului dintre salarii şi preţuri:

Salarii ⟶ Preţuri

Buclele feedback pot fi pozitive şi negative. O buclă feedback pozitivă este aceea care, dacă este parcursă complet, conduce la modificarea variabilelor în aceiaşi direcţie. De exemplu, creşterea salariilor duce la creşterea preţurilor în timp ce scăderea salariilor determină scăderea preţurilor. Deci bucla feedback dintre preţuri şi salarii este pozitivă.

O buclă feedback negativă este aceea în care o creştere a unei variabile din cadrul buclei duce după parcurgerea în întregime a buclei, la descreşterea variabilei respective. De exemplu, o buclă feedback dintre numărul de angajări şi volumul vânzărilor se poate reprezenta astfel:

Număr de angajări ⟶ Volumul vânzărilor

În sistemele complexe, de multe ori, buclele feedback care apar sunt multiple, deci formate din două sau mai multe bucle feedback de polarităţi diferite. Datorită influenţelor pe care le exercită, aceste bucle feedback trebuie analizate astfel încât să se determine care dintre ele este dominantă. O buclă feedback dominantă este acea buclă ce determină în cea mai mare măsură tendinţa de modificare a variabilelor incluse în conturul buclelor respective. În exemplul următor avem două bucle feedback, una negativă şi cealaltă pozitivă care acţionează concomitent. Fig. 3.2

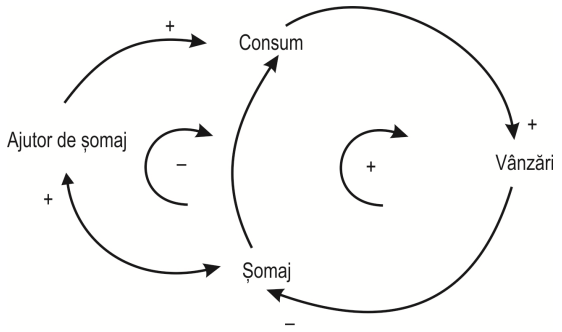


Figura 3.2

Este clar aici că bucla feedback pozitivă este dominantă, cealaltă buclă feedback influenţând consumul doar într-o măsură redusă, existând alte mărimi care determină într-o măsură mai mare consumul, de exemplu venitul din salarii.

Aceste bucle feedback multiple constituie de fapt, regula în sistemele economice reale. De regulă, determinarea structurii buclelor feedback dintr-un sistem economic real se realizează într-un proces iterativ, pornind mai întâi de la buclele feedback evidente şi importante, după care se adaugă noi bucle feedback până când se ajunge la structură corespunzătoare, care să permită obţinerea unui model cât mai complet.

După aceste două etape, care constituie partea calitativă a analizei, urmează etapele ce conduc la analiza cantitativă, absolut necesară pentru a obţine, în final, un model valid şi util în simulare.

3.2.3. Formularea şi analiza modelului

Am arătat că există două tipuri importante de fluxuri distincte în modelele de dinamică de sistem: fluxuri materiale şi fluxuri informaţionale. Aceste două tipuri de fluxuri se introduc în mod diferit în modele.

Fluxurile materiale şi întârzierile

Fluxurile materiale sunt cele care transferă dintr-un punct în altul al sistemului produse, oameni, bani, comenzi etc. Aceste fluxuri au o caracteristică importantă, şi anume, prezintă întârzieri, deci timpul necesar efectuării transferurilor respective pot depăşi o perioadă de timp standard în modelul respectiv (notat cu DT şi care poate fi o zi, o săptămână, o lună sau un an). De exemplu, dacă într-un model apare evoluţia populaţiei dintr-o economie, se poate utiliza o ecuaţie de forma:

$$\mathbf{POP_t = POP_{t-1} + DT\ (naşteri_{dt} - decese_{dt})}$$

care spune că populaţia de la momentul t este egală cu populaţia de la momentul t-1, la care se adaugă diferenţa dintre numărul de naşteri între t şi t-1 (deci DT) şi numărul de decese în acelaşi interval DT. În principiu, aceasta este ecuaţia de bază pentru orice tip de stoc ce ar putea conţine oameni, produse, bani, comenzi etc.

Stocurile (nivelele) din sistem pot fi identificate imaginându-ne că fluxurile se opresc şi se acumulează în nivele o perioadă de timp egală cu DT. Valorile acestor stocuri, la un moment de timp dat, reprezintă o „imagine statică" a sistemului sau o valoare a vectorului de stare la acel moment de timp. Dacă asociem ecuaţiei de nivel a evoluţiei populaţiei o diagramă cauzală, observăm că populaţia este influenţată de două bucle feedback, amândouă pozitive (figura 3.3).

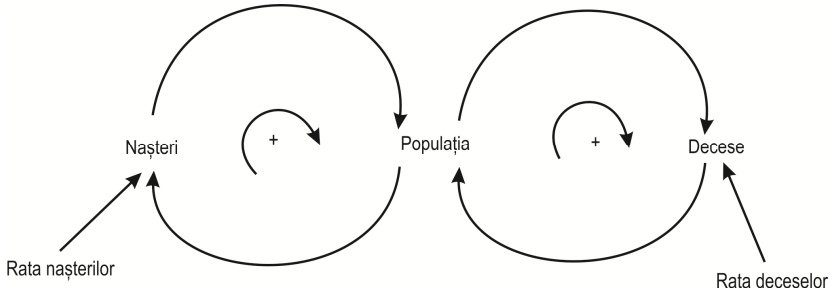


Figura 3.3

Această ecuaţie poate fi reprezentată şi printr-o diagramă de flux ca în figura 3.4.

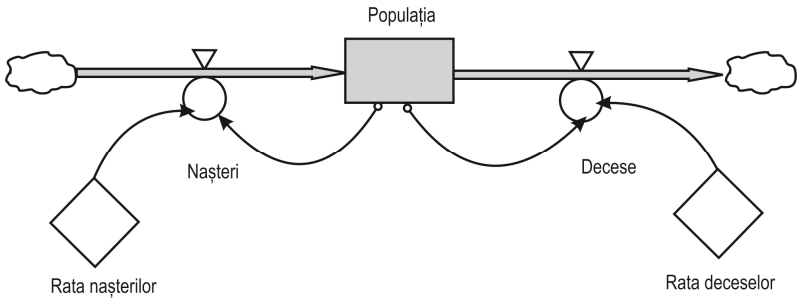


Figura 3.4

Cele două reprezentări sunt echivalente. De regulă, diagramele cauzale sunt utilizate în analiza preliminară a sistemului în timp ce diagramele de flux sunt utilizate în stadiul trecerii la modelul formal al sistemului. Se observă din a doua diagramă că orice stoc funcționează ca un integrator deoarece acumulează fluxurile de la intrare. Fiecare ecuaţie de nivel conţine şi intervalul de timp de simulare, DT. Trebuie făcută o distincţie clară între doua puncte: (a) timpul de simulare; (b) intervalul de timp al modelului; şi (c) incrementul DT. Timpul de simulare sau orizontul de timp este perioada de timp totală în care modelul este simulat. Intervalul de timp este specificat de cel care construieşte modelul şi poate fi egal cu ani, luni, săptămâni sau zile, în funcţie de scopurile urmărite. De exemplu, într-un model al populaţiei nu are sens ca acest orizont de timp să fie egal cu zile sau săptămâni.

Intervalul de timp arată momentele de timp în care modelul furnizează valori calculate ale nivelelor. Dacă dorim să studiem evoluţia populaţiei pe o perioadă de 20 de ani (orizontul de timp) atunci acest interval de timp poate fi luat egal cu un an.

În sfârşit, incrementul de timp DT specifică numărul de cicluri de calcul de-a lungul unui interval de timp al modelului. Dacă DT este stabilit la 1, iar intervalul de timp al modelului este de un an, atunci modelul calculează mărimea nivelului (de exemplu populaţia) o dată pe an. Dacă DT este stabilit la 1/12 atunci vor fi 12 cicluri de calcul ale populaţiei de-a lungul unui an, deci câte un ciclu de calcul în fiecare lună.

Alegerea lui DT se face după anumite reguli. Cu cât există mai multe cicluri de calcul în cadrul unui interval de timp, cu atât soluţia obţinută este mai exactă. Dar creşterea numărului de cicluri, altfel spus stabilirea pentru DT a unei valori foarte mici duce la creşterea timpului de simulare care, uneori, trebuie să fie în concordanţă cu timpul necesar pregătirii deciziilor ce urmează să fie adoptate în sistem.

Deci ecuaţiile de nivel au o structură standard, de forma:

$$\mathbf{NIVEL_t = NIVEL_{t-1} + DT(RATA\ INTRARE_{dt} - RATA\ IESIRE_{dt}),}$$

Ecuaţiile de ritm nu au o structură standard. Totuşi, analizând atent factorii care determină cunoaşterea nivelelor, respectiv reducerea acestora, se pot stabili formele matematice ale acestor ecuaţii de ritm. De exemplu, este clar că numărul de naşteri depinde de nivelul populaţiei şi de rata naşterilor. Putem scrie atunci că:

Numărul de naşteri = Nivelul Populaţiei × Rata Naşterilor

La fel pentru decese avem,

Numărul de decese = Nivelul Populaţiei × Rata Deceselor

Cele două rate controlează fluxul de intrare, respectiv fluxul de ieşire din nivelul populaţiei. Funcţionarea lor concomitentă va determina creşterea nivelului populaţiei dacă prima rată este mai mare sau dimpotrivă, scăderea numărului populaţiei dacă a doua rată este mai mare.

Evoluţia valorilor calculate pentru nivelele din cadrul unui model de dinamică de sistem se reprezintă grafic (cu ajutorul unor proceduri grafice existente în orice limbaj de simulare). Aceste reprezentări grafice arată vizual ce se întâmplă cu nivelul respectiv de-a lungul orizontului de simulare considerat. Forma curbelor obţinute poate da o idee asupra timpului de buclă feedback în care se află nivelul respectiv sau a raportului de dominantă dintre buclele feedback respective.

În figura 3.5 se reprezintă două grafice ale evoluţiei populaţiei, obţinute primul dintr-o buclă feedback pozitivă (creşterea este continuă şi nelimitată), iar al doilea dintr-o buclă feedback negativă (deşi avem creştere, aceasta este plafonată).

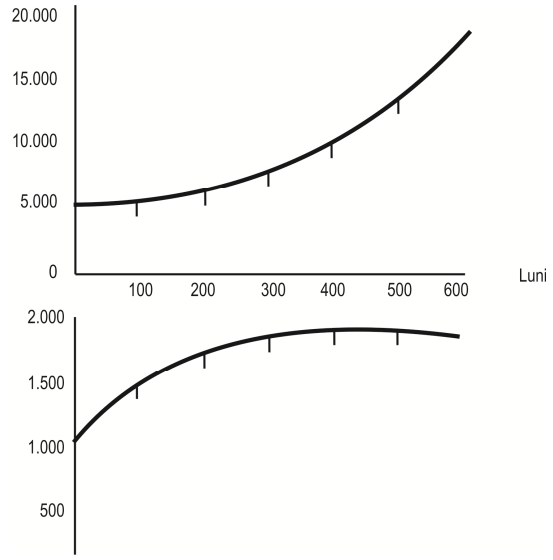


Figura 3.5

De regulă, graficele reprezentând evoluţia nivelelor se reprezintă în acelaşi sistem de axe de coordonate, ceea ce face posibilă compararea şi analiza corelaţiilor existente între ecuaţiile diferitelor nivele ale modelului.

În modelele dinamică de sistem apar reprezentate mai multe tipuri de fluxuri. Forrester a introdus şase tipuri de astfel de fluxuri: oameni, bunuri, comenzi, bunuri de capital, bani şi fluxuri informaţionale. Cu excepţia fluxurilor informaţionale, toate celelalte sunt fluxuri materiale. Ele se comportă identic în ceea ce priveşte stocurile (nivelele) şi a celorlalte elemente care intervin în model: ecuaţiile de ritm, constantele şi întârzierile. Incrementul de timp DT poate fi ales acelaşi pentru toate aceste fluxuri materiale. O diferenţă care trebuie, totuşi, făcută în aceste fluxuri materiale este cea dintre întârzierea propagării lor în sistem. Astfel, dacă ne referim la fluxul de bunuri şi la fluxul de capital, este clar că bunurile rămân în sistem mult mai puţin timp decât capitalul. Primele sunt produse, stocate şi livrate în acelaşi interval de timp. Spunem că întârzierea este, în acest caz, egală cu 1 (DELAY 1). Bunurile capitale (maşini, instalaţii, infrastructura destinată producţiei) rămân în sistem o perioadă mult mai lungă, de regulă, până când ele sunt complet amortizate şi chiar după aceea.

Avem deci o întârziere de tip $DELAY_n$, unde n se stabileşte în raport cu tipul de bunuri capitale utilizate. O altă diferenţă între întârzieri se manifestă, să spunem, între fluxul de oameni şi fluxul de bani. Oamenii (angajaţii) pot rămâne în sistem o perioadă îndelungată în timp ce banii au o perioadă de staţionare mult mai scurtă.

Această diferenţă între întârzierile ce afectează fluxurile de natură diferită se introduce în modelele de dinamică de sistem prin legarea în serie a mai multor nivele în cazul fluxurilor cu întârzieri mari. Pentru a nu complica prea mult modelul, atunci când avem o astfel de situaţie, în cadrul nivelului se specifică acest lucru. De exemplu, dacă avem un flux întârziat cu trei perioade, acest lucru se reprezintă ca în figura 3.6:

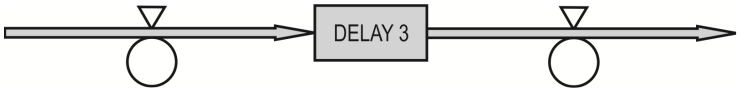


Figura 3.6

Mărimea capitalului K va rămâne în stoc trei perioade de timp până când ritmul de ieşire va afecta nivelul respectiv.

Fluxurile informaţionale şi întârzierile lor

Fluxurile informaţionale joacă un rol important în modelele de dinamică de sistem. Informaţia leagă între ele diferitele fluxuri din model. Ea este utilizată în pregătirea politicilor decizionale. De exemplu, mărimile de stare (nivele) calculate pe diferite fluxuri materiale furnizează informaţia pentru adoptarea deciziilor care influenţează ale fluxuri materiale din model. Stocul de bani existent influenţează deciziile privind angajarea de noi muncitori sau achiziţia de bunuri destinate producţiei. Rezultă deci că aceste fluxuri informaţionale împreună cu politicile decizionale în care intervin reprezintă mecanismul de control al sistemului. De regulă, politicile decizionale se manifestă prin intermediul ratelor care intervin pe fluxurile materiale. O astfel de politică decizională constă din patru elemente: un scop (sau o stare dorită), o stare observată a sistemului, o diferenţă calculată între starea dorită şi cea observată şi o acţiune bazată pe această diferenţă. Să considerăm, de exemplu, diagrama de flux din figura 3.7.

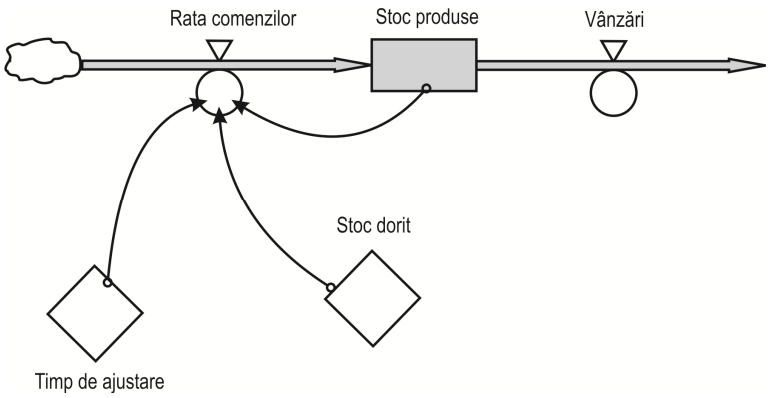


Figura 3.7

Se observă că stocul de produse este redus prin vânzări. Decizia de a comanda noi produse depinde de diferenţa dintre Stoc Dorit şi nivelul aflat în Stoc Produs. Dar acest lucru poate induce în sistem o instabilitate deoarece stocul poate fluctua de la o perioadă la alta, determinând ca Rata Comenzilor să ia valori fluctuante. De aceea se introduce o constantă denumită Timp Ajustare care exprimă timpul necesar eliminării diferenţei constante între stocul dorit şi stocul existent. Cu cât această constantă este mai mare, cu atât rata comenzilor este mai puţin fluctuantă.

Rata Comenzilor = (Stoc Dorit – Stoc Produse)/Timp de Ajustare

Se observă că dacă Stocul Dorit este egal cu stocul existent, Rata Comenzii este egală cu zero. Cu cât diferenţa dintre ele este mai mare cu atât Rata Comenzii va fi mai mare, dar nu va fi egală cu valoarea diferenţei respective ci cu o fracţiune din ea, dată de mărimea timpului de ajustare.

Evident că aceste mecanisme asociate politicilor decizionale pot fi mult mai complicate, incluzând mult mai multe conexiuni informaţionale astfel încât controlul exercitat asupra variabilelor de stare ale sistemului să permită atingerea scopurilor urmărite.

Prin intermediul fluxurilor informaţionale se introduc în model şi diferite funcţii standard (funcţia rampă, funcţia treaptă, funcţia impuls, funcţia aleatoare etc.) urmărindu-se apoi care este răspunsul sistemului la diferite şocuri şi perturbaţii ce pot apărea în mediul înconjurător. Aceste funcţii sunt utilizate mai ales în etapele de testare şi evaluare a modelului pentru a aprecia senzitivitatea acestuia, deci răspunsul pe care îl dă modelul la diferite perturbaţii deterministe sau aleatoare.

3.2.4. Formularea modelului

În continuare, un pas important îl reprezintă formularea modelului de dinamică de sistem. Acest lucru presupune mai întâi formularea ecuaţiilor, verificarea consistenţei acestora şi estimarea parametrilor care apar în model.

Formularea ecuaţiilor şi verificarea consistenţei acestora

După cum am văzut mai sus, structura ecuaţiilor de nivel este bine stabilită şi similară pentru toate nivelele. Limbajele de simulare cum ar fi Stella, Powersim sau Vensim produc automat ecuaţiile de nivel care apar în diagrama de flux. Totuşi, ecuaţiile de ritm şi ecuaţiile auxiliare nu sunt standardizate, ele trebuind să fie elaborate şi introduse în model de către constructorul de modele. De exemplu, în modelul de control al stocurilor, ecuaţia comenzilor se scrie:

$$\text{comenzi} = (\text{stoc dorit} - \text{stoc}) \times \frac{1}{\text{timp de ajustare}}$$

Se observă, totuşi, că o astfel de ecuaţie funcţionează doar dacă stocul dorit este mai mare decât stocul existent. În caz contrar, comenzile vor fi egale cu zero, neputând fi egale cu valori negative. Se poate, atunci, introduce o funcţie MAX astfel că ecuaţia de mai sus devine:

$$\text{comenzi} = \text{MAX}(0, \left((\text{stoc dorit} - \text{stoc}) \times \frac{1}{\text{timp de ajustare}} \right))$$

Ecuaţiile din model sunt verificate în ceea ce priveşte consistenţa lor internă. Verificarea presupune înlocuirea variabilelor din ecuaţii cu dimensiunile lor. Dacă prin această operaţie, ecuaţia se dovedeşte validă, atunci ea se numeşte „consistentă dimensional". În cazul ecuaţiei de ritm de mai sus, comenzile se măsoară prin [Articole / Săptămână], stocurile dorite şi curente se măsoară prin [Articole], iar timpul de ajustare se măsoară în [Săptămâni]. Înlocuind acum aceste dimensiuni în ecuaţia de ritm obţinem:

$$[\text{Articole} / \text{Saptamana}]$$
$$= \text{MAX}(0, ([\text{Articole}] - [\text{Articole}]) \times \frac{1}{[\text{Saptamana}]})$$

După reducerile evidente, obţinem:

$$[\text{Articole} / \text{Saptamana}] = [\text{Articole} / \text{Saptamana}]$$

deci ecuaţia considerată este consistentă dimensional.

Consistenţa dimensională poate fi verificată şi în cazul ecuaţiilor de nivel. De exemplu, ecuaţia de nivel a populaţiei:

$$\text{POP}_t = \text{POP}_{t-1} + \text{DT}(\text{rata nasterilor}_{dt} - \text{rata deceselor}_{dt})$$

are dimensiunile corespunzătoare următoare:

$$[\text{P}] = [\text{P}] + [\text{T}] \times \left(\left[\frac{\text{P}}{\text{T}} \right] - \frac{\text{P}}{\text{T}} \right) \Rightarrow$$

$$[\text{P}] = [\text{P}] + [\text{T}] \times \left[\frac{\text{P}}{\text{T}} \right]$$

$$[\text{P}] = [\text{P}] + [\text{P}]$$

$$[\text{P}] = [\text{P}]$$

deci ecuaţia este consistentă dimensional.

Consistenţa dimensională se verifică automat de către limbajele de simulare menţionate. În cazul în care se constată că anumite ecuaţii nu sunt consistente dimensional ele sunt corectate astfel încât să respecte acest criteriu important. Dacă şi numai dacă toate ecuaţiile modelului sunt verificate şi se dovedeşte că este consistent intern, el poate fi utilizat în continuare în celelalte etape.

Estimarea parametrilor

Multe dintre ecuaţiile care intervin în modelele de dinamică de sistem conţin unul sau mai mulţi parametri. De exemplu, valorile iniţiale ale variabilelor de nivel constituie o mulţime importantă de parametri. Populaţia iniţială sau stocul la momentul zero sunt exemple de astfel de parametri.

O altă categorie de parametri sunt cei care intervin în ecuaţii şi au valori constante: stocul dorit, timpul de ajustare, rata naşterilor, rata deceselor etc. sunt considerate constante ale modelului.

Există mai multe metode de determinare a valorilor parametrilor din modelele de dinamică de sistem. Cea mai frecvent aplicată metodă este cea de cunoaştere directă a procesului. Prin culegerea datelor privind evoluţia reală a procesului se pot determina valori ale constantelor.

Alte metode utilizate sunt cele statistice, deoarece unele constante reprezintă valori medii calculate pe baza unor date observate. Totuşi, dacă procesele sunt complicate şi nu există suficiente date pentru estimarea parametrilor, se poate utiliza o metodă de simulare cum ar fi metoda Monte Carlo. Aceasta se poate realiza doar în condiţiile în care dispunem de anumite informaţii privind distribuţia de probabilitate a datelor culese din procesele reale.

Parametrii trebuie şi ei exprimaţi în unităţi dimensionale corespunzătoare care sunt luate în considerare la verificarea consistenţei dimensionale. Alt aspect priveşte acurateţea estimării parametrilor. În legătură cu efortul necesar pentru a creşte acurateţea determinării parametrilor se utilizează următoarea regulă:

Dacă implicaţiile politice ale unui model nu se schimbă dacă parametrii acestuia se schimbă cu plus sau minus un procent, atunci valorile parametrilor nu este necesar să se mai modifice printr-o estimare suplimentară. Această problemă, împreună cu alte aspecte pentru asigurarea acurateţei întregului model se rezolvă în următoarea etapă a elaborării modelelor de dinamică de sistem.

3.2.5. Testarea modelului şi analiza de senzitivitate

Aceasta reprezintă un pas important în realizarea modelelor de dinamică de sistem. Modelele de simulare permit testarea repetată în urma căreia creşte înţelegerea acestora. Să arătăm, în continuare, cum se realizează analiza de senzitivitate pentru un model de control al stocurilor cu întârziere în informaţie. În figura 3.8 se prezintă rezultatele simulării unui astfel de model în care DT este egal cu o săptămână, iar orizontul de simulare este de 80 de săptămâni. Fiecare

curbă reprezentată corespunde unui timp de percepţie diferit. Astfel, curba 1 corespunde unui timp de percepţie de 4 săptămâni, curba 2 corespunde unui timp de percepţie de 6 săptămâni, iar curba 3 corespunde unui timp de percepţie de 8 săptămâni.

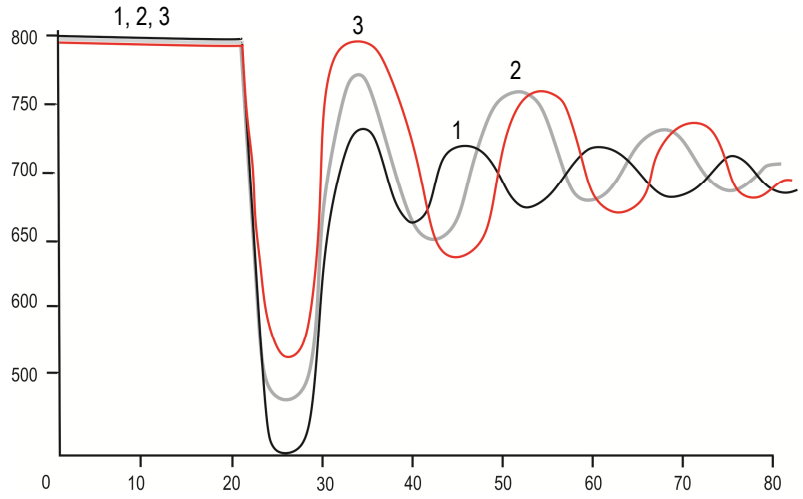


Figura 3.8

Se observă că indiferent de timpul de percepţie, după 80 de săptămâni se ajunge la o stabilizare a traiectoriilor în jurul valorii de 700. Fiecare traiectorie 1, 2 sau 3 are, însă amplitudini diferite ale fluctuaţiilor. Altfel spus, dublarea timpului de percepţie duce la creşterea amplitudinii fluctuaţiilor nivelului stocului dar nu schimbă rezultatul final.

Un alt test de senzitivitate poate consta într-o schimbare a nivelului iniţial al stocului, a timpului de ajustare sau a stocului dorit. Pentru a fi eficiente, aceste teste trebuie să fie aplicate sistematic, adică să se efectueze pentru fiecare parametru menţionat considerând ceilalţi parametri constanţi.

De regulă, aceste teste se efectuează conform unei metodologii care include reguli bine stabilite. Astfel, o primă regulă este să se elaboreze o listă de teste care se doreşte să fie efectuate. O a doua regulă este să se efectueze teste separate atunci când un model este mărit prin adăugarea unor noi componente sau subsisteme. Aceste teste parţiale sunt apoi completate cu testarea întregului model, urmărindu-se dacă noua componentă introdusă răspunde la fel în ambele tipuri de teste. O a treia regulă priveşte predicţia efectului testelor înainte de rularea efectivă a modelului. În cazul în care rezultatele testelor se îndepărtează prea mult de rezultatele prognozate atunci se poate concluziona că există anumite probleme în model care trebuie corectate înainte de utilizarea lui efectivă.

Testele se efectuează, de regulă, după ce se efectuează o rulare standard cu modelul. Această rulare standard serveşte în continuare ca bază de comparaţie. Pe măsură ce se efectuează testele, diferenţele constatate între rezultatele fiecărui test şi rezultatul din rularea standard sunt explicate în raport cu modificările aduse parametrilor şi ecuaţiilor din model.

Un test important este cel prin care una sau mai multe bucle feedback sunt dezactivate. În acest fel, se poate urmări efectul exact al unei bucle feedback asupra comportamentului întregului model.

Analiza de senzitivitate îndeplineşte două funcţii importante. Prima este aceea că aceste teste de senzitivitate servesc la mai buna înţelegere a modelului. A doua funcţie este că prin testele de senzitivitate se urmăreşte localizarea parametrilor de senzitivitate din model. Se răspunde astfel să se determine cât de senzitiv este comportamentul modelului la schimbarea valorilor parametrilor. În general, mulţi parametri din modelele de dinamică de sistem sunt relativ insensibili la schimbările parametrilor în limite rezonabile. Primul motiv pentru aceasta este acela că se poate schimba un parametru care nu este conţinut într-o buclă feedback. Un al doilea motiv este că buclele feedback pot să se compenseze una pe cealaltă. De exemplu, dacă schimbăm un parametru dintr-o buclă feedback pozitivă care este legată cu o buclă feedback negativă, acest lucru poate duce la compensarea efectului buclei feedback negative prin această schimbare.

Dacă se identifică un parametru senzitiv într-un model, în continuare se poate proceda în trei moduri: Primul este că se poate decide să se estimeze acel parametru cât mai atent posibil deoarece acesta va avea un efect important asupra comportamentului modelului. Al doilea, se poate decide să se rafineze modelul pentru a surprinde procesul agregat în care parametrul este inclus în cele mai mici detalii. Al treilea, dacă senzitivitatea parametrului corespunde senzitivităţii din sistemul real se poate identifica un punct prin care întregul model să poată influenţa comportamentul sistemului.

3.2.6. Evaluarea modelului

O altă etapă importantă în construirea modelului este cea de evaluare. Prin aceasta se urmăreşte adecvanţa modelului în relaţie cu reprezentarea sistemului. De regulă, această adecvare se reduce la studierea validităţii modelului. La prima vedere, validitatea priveşte capacitatea modelului de a reproduce comportamentul sistemului. Un model absolut valid nu există. Acest lucru ar presupune că reproducerea comportamentului sistemului este exactă. Datorită complexităţii sistemelor reale întotdeauna apare o reprezentare simplificată în cazul unor comportamente.

Un lucru important este cel al raportului dintre validitate şi utilitate. Dacă trebuie investiţi mai mulţi bani şi timp pentru ca modelul să fie valid fără ca utilitatea acestuia să crească, atunci trebuie să se renunţe la validitate în favoarea utilităţii.

O altă premisă este aceea că, în condiţiile în care validitatea exactă nu există în realitate, trebuie sporit gradul de confidenţă în model. Creşterea gradului de confidenţă este un proces care se construieşte treptat şi care nu ia sfârşit nici atunci când modelul este terminat.

Pentru a construi gradul de confidenţă, modelul este supus unui număr de teste. Cu cât numărul de teste este mai mare, cu gradul de confidenţă este mai mare. În diferite lucrări se recomandă un număr de 17 teste diferite pentru validarea modelului. Două dintre aceste teste privesc structura modelului. Un prim test în această privinţă este cel al verificării consistenţei dimensionale a modelului. Dacă modelul nu trece acest test atunci el este considerat nefolositor deoarece nu este clar în ce dimensiuni se măsoară output-ul modelului. În plus, toţi parametrii modelului este necesar să aibă o interpretare în lumea reală.

Un alt test îl reprezintă validitatea formei modelului. Modelele de dinamică de sistem au o structură care este asemănătoare structurii sistemului în privinţa subsistemelor componente, a fluxurilor conţinute şi a mecanismelor decizionale exacte. Nu poţi introduce în model subsisteme care să nu aibă corespondent în realitate sau fluxuri de altă natură decât cele din sistemele reale. Dacă mecanismele decizionale din realitate sunt mai greu de identificat, totuşi deciziile adoptate în ambele sisteme, cel real şi cel modelat, sunt de aceeaşi natură. Această corespondenţă dintre structura sistemului real şi structura modelului de dinamică de sistem reprezintă o condiţie de bază în asigurarea adecvanţei modelului la realitate.

Alte teste sunt legate de comportamentul modelului. Am văzut mai sus că prin analiza de senzitivitate se urmăreşte ca modelul să nu producă comportamente prea senzitive la schimbări mici în parametri. Un alt test de aceeaşi natură priveşte analiza de senzitivitate a modelului la condiţii extreme. În mediul înconjurător al sistemului se pot întâlni cazuri în care apar schimbări dramatice care trebuie să producă în model comportamente corespunzătoare. De exemplu, în modelul de control al stocurilor poate apărea situaţia în care rata comenzilor nu este suficient de mare şi duce la o rupere de stoc. În acest caz, apare o scădere a stocului existent sub valoarea zero, lucru care în sistemele reale nu este posibil (evident că acest aspect se referă la o lipsă de articole în stoc ce are efecte dramatice asupra altor componente ale sistemului).

Pentru a evita astfel de lucruri, modelul de control al stocurilor trebuie îmbunătăţit prin introducerea unui mecanism care să perceapă din timp posibilitatea de apariţie a ruperilor de stoc. O astfel de modificare a modelului este prezentată în figura 3.9.

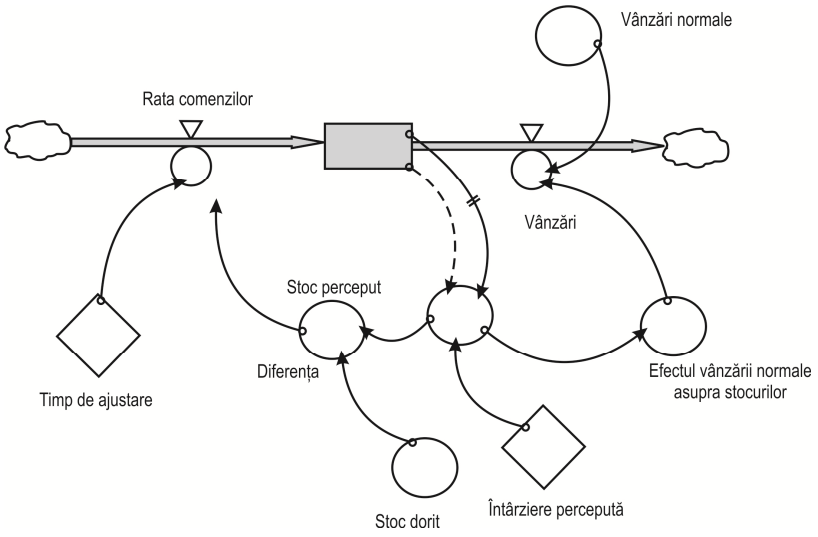


Figura 3.9

În această variantă a modelului vânzările curente sunt înlocuite cu vânzări normale, care sunt considerate exogene. Acestea sunt ajustate cu un coeficient și dau efectul vânzărilor normale asupra stocurilor, care exprimă ce nivel al stocului trebuie menținut astfel ca vânzările să nu fie afectate. Astfel, dacă stocurile scad se va acționa asupra ritmului vânzărilor, reducându-l astfel încât nivelul stocului să fie menținut pozitiv. Pe de altă parte, dacă nivelul stocurilor este suficient de mare atunci toate livrările vor fi efectuate, crescând ritmul vânzărilor. Acest lucru se realizează cu ajutorul unei funcții GRAPH, care specifică relaţia dintre nivelul stocului şi efectul acestuia asupra vânzărilor. O astfel de funcție GRAPH este reprezentată în figura 3.10.

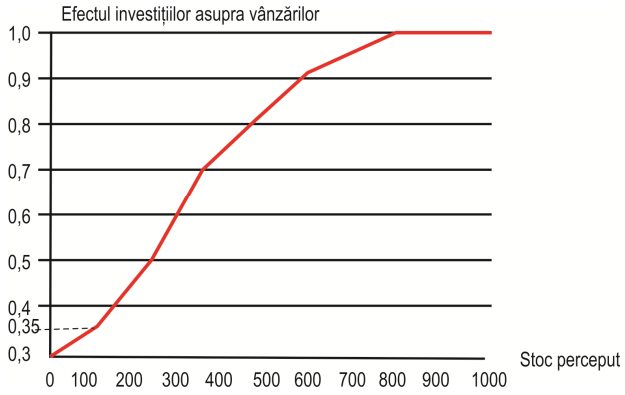


Figura 3.10

De exemplu, dacă nivelul stocului este redus, să spunem 100, vânzările vor fi reduse cu un factor de multiplicare de 0,35. În schimb, dacă nivelul stocului este mare, de exemplu peste 700 unități, atunci vânzările sunt mult mai puțin afectate, să spunem cu un factor de 0,9. O dată de stocul depășește 800 de unități, vânzările nu sunt afectate, factorul de multiplicare fiind luat egal cu 1.

Prin introducerea unui astfel de mecanism, oscilațiile în nivelul stocului sunt mult mai reduse, acestea ajungând repede la un nivel stabilizat (vezi figura 3.11).

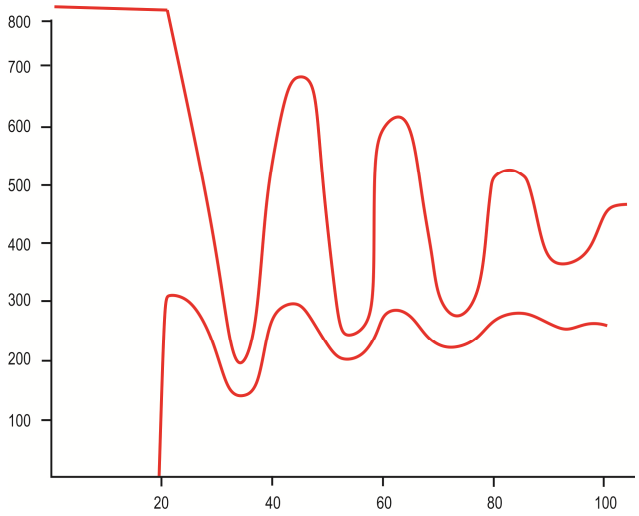


Figura 3.11

Exemplul de mai sus arată că testele de validitate pot contribui la îmbunătățirea structurii modelelor astfel încât să crească încrederea în model.

3.2.7. Analiza politică

Cel mai important obiectiv urmărit prin construirea unui model de dinamică de sistem este cel de a testa diferite politici destinate îmbunătățirii performanței sistemului. Analiza politicilor decizionale se orientează către schimbarea punctelor în care se adoptă decizii, precum și asupra efectelor pe care le au acestea asupra variabilelor rezultative. Schimbările în funcțiile de decizie pot include schimbări ale parametrilor și schimbări structurale. În primul caz, analiza politică presupune o schimbare a valorii numerice a unui parametru. De exemplu în modelul de control al stocului, se poate schimba valoarea întârzierii în percepția

stocului și se poate urmări ce se întâmplă cu evoluția nivelului stocului pentru diferite valori ale acestui parametru.

Mai simplificată este analiza politică în cazul schimbărilor structurale ale modelului. Acest lucru presupune încorporarea în model a unor noi fluxuri informaționale, formarea de noi bucle feedback, sau chiar realizarea de noi componente care sunt integrate modelului. Pentru fiecare modificare ce se realizează, se generează traiectorii care reprezintă comportamentele sistemului în respectivele condiții. De fiecare dată, aceste rezultate sunt comparate între ele urmărindu-se să se obțină acea traiectorie care se apropie cel mai mult de comportamentul dorit.

În acest context se vorbește și de politica robustă. O astfel de politică robustă este aceea care nu se modifică semnificativ dacă au loc în continuare schimbări structurale ale modelului. Robustețea unei politici este uitată deoarece nici un model nu este identic cu sistemul real. De fapt, în această privință circulă zicala conform căreia: „toate modelele sunt greșite, dar unele dintre ele sunt mai utile decât altele". Acest lucru are consecințe importante asupra robusteții politicilor recomandate ale modelelor respective, în sensul că cele mai utile modele sunt cele care recomandă politici decizionale robuste. Scopul analizei politice este tocmai acela de a descoperi politici robuste utilizând modelele de dinamică de sistem.

3.2.8. Utilizarea sau implementarea modelului

Ultima etapă în procesul de elaborare a modelului de dinamică de sistem, este trecerea efectivă la utilizarea modelului. Dacă modelul a arătat că o anumită politică produce o îmbunătățire semnificativă și că această politică este robustă, atunci un pas firesc este implementarea unei astfel de politici.

În literatură există nenumărate exemple ale unor modele de dinamică de sistem. Prin acumularea de experiență în acest domeniu s-au elaborat mai multe ghiduri de implementare (vezi: **Roberts**, 1978; **Weil**, 1980; **Lane**, 1992). Fiecare dintre aceste ghiduri conțin recomandări concrete asupra pașilor care trebuie urmați în vederea implementării unui astfel de model.

Într-un astfel de ghid, elaborat de Roberts se arată că orice model implementat trebuie să rezolve o anumită problemă bine specificată, definită de client. Acest lucru implică faptul că în cursul etapei de implementare constructorul de modele trebuie să comunice permanent cu clientul pentru a fi sigur că a înțeles care sunt problemele acestuia. O altă recomandare importantă, ce derivă din recomandarea precedentă, este aceea că clientul trebuie să se implice în procesul de construire a modelului cât mai mult posibil.

Weil, compară trei procese de implementare a trei modele de dinamică de sistem și stabilește câteva reguli de care se ține seama în etapa de implementare. El observa că în proiectele implementate într-o etapă timpurie a dinamicii de

sistem, accentul cade pe model, existând o preocupare exagerată de a crea modele cât mai credibile. În proiectele implementate mai recent se pune accentul pe client și pe cerințele acestuia, client care era tot mai implicat în procesul de realizare a modelului. În plus se remarcă creșterea interesului pentru transferul de know-how, pentru a crea la nivelul organizației o „capacitate analitică", în ceea ce privește utilizarea rezultatelor modelelor.

O concepție diferită asupra implementării modelelor de dinamică de sistem introduce **Lane**, care vorbește despre învățare ca o metodă profund implicată în procesul de modelare. Învățarea are loc în toate etapele de elaborare ale modelului ca atare.

3.3. Simularea în modelele de dinamică de sistem

3.3.1. Principii generale

Ideea esențială în simularea modelelor de dinamică de sistem este aceea a determinării valorilor nivelelor în fiecare dintre momentele de timp dintr-un orizont dat. Aceste momente în care se determină valorile nivelelor se stabilesc utilizând valoarea DT care este specificată pentru fiecare model. De exemplu, dacă DT = 0,1 iar orizontul de timp este egal cu 100, vor exista 100/0,1 = 1000 momente de timp în care vom determina valorile nivelelor. Aceste valori reprezintă de fapt valorile unor variabile de stare (asociate nivelelor), succesiunea lor în timp pe întregul orizont T dat reprezentând traiectoria de evoluție a sistemului dinamic modelat.

Pentru a putea determina nivelele sunt necesare următoarele elemente:

- Valorile nivelelor la momentul de timp inițial (condițiile inițiale). Acest moment inițial este considerat de cele mai multe ori zero. Dacă nu este zero se poate realiza o translație pe axa timpului astfel încât momentul inițial să fie zero;
- Ritmurile asociate intervalelor de timp, care separă momentele în care sunt calculate nivelele;
- Un mijloc de calcul al integrării ritmului pentru a furniza valoarea următoare a nivelului;
- Ritmurile vor depinde, în general, doar de nivele și auxiliare, sau în cazul ieșirii dintr-o întârziere, de valorile trecute a acestui ritm pe perioada întârzierii;
- Ecuațiile auxiliare sunt doar etape intermediare în calcularea unei ecuații de ritm pentru ca aceasta să aibă o formă mai simplă. Ele vor depinde de nivele și eventual de alte auxiliare;

Dacă într-o secvență de calcul intervin mai multe auxiliare, de exemplu:

AUX2.K = AUX1.K + NIVEL.K/DEL

atunci AUX1.K trebuie întotdeauna calculat înainte de AUX2.K;

Într-o secvență de calcul nu trebuie să avem ritmuri care depind unele de altele, de exemplu:

RITM1.KL = AUX1.K+RITM2.KL

RITM2.KL = 2*RITM1.JK

deoarece se formează o buclă fără nivele, ce produce dinamici false și dependente doar de valorile numerice ale lui DT. O astfel de secvență se numește neexecutabilă și este identificată automat de limbajul de simulare modelare.

Diferența dintre două puncte în care se efectuează un ciclu executabil complet al modelului, (deci se calculează toate nivelele corespunzătoare precum și ritmurile valabile între cele două puncte) se numește **interval de simulare** și este egal cu DT unități de timp. Perioada scursă de la momentul inițial 0 până la timpul prezent se notează cu TIME, iar durata întregii simulări se notează cu LENGH. Evident că dacă împărțim TIME sau Lengh la valoarea lui DT obținem pașii de simulare executați până la momentul prezent, respectiv numărul total de pași de simulare pe care îi parcurge modelul.

Procedura de simulare pe care se bazează toate limbajele utilizate în dinamica de sistem constă din aplicarea următoarelor reguli de bază:

- Deoarece nivelele depind de ritmuri și nu de alte nivele, valorile lor curente se pot determina în orice ordine;
- Ritmurile și auxiliarele se calculează pe baza valorilor determinate anterior ale nivelelor;
- O simulare completă poate să includă multe momente de timp, dar trei dintre acestea sunt esențiale. Ele se notează cu J, K și L, iar perioadele de timp dintre ele cu JK și KL, fiecare dintre aceste perioade fiind egale cu DT.

În figura 3.12 se prezintă procedura generală de calcul aplicabilă pentru fiecare moment de timp. Punctul K este corespunzător valorii curente și o dată calculate nivelele la momentul K vor fi determinate auxiliarele la momentul K și ritmurile pentru intervalul KL. Dacă nu s-a ajuns la sfârșitul intervalului de simulare atunci se avansează cu un moment de timp, K devenind J, L devenind K, iar intervalul KL devenind JK.

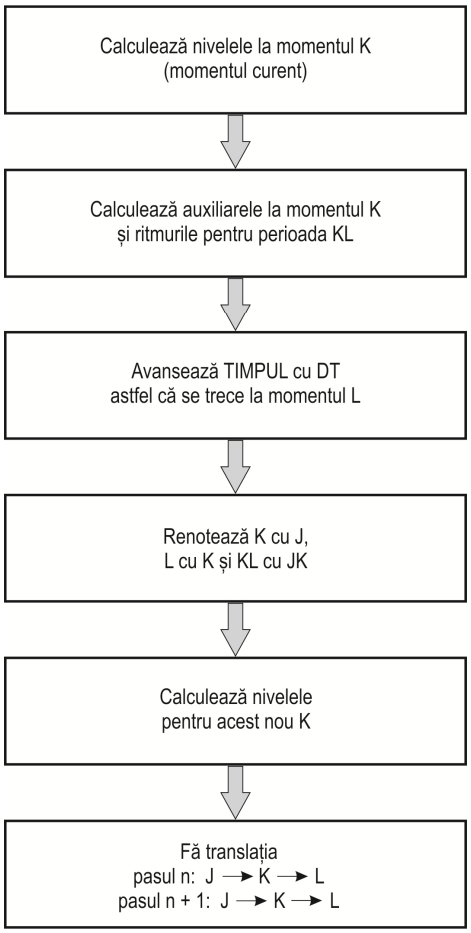


Figura 3.12

Cu cât timpul avansează, valorile calculate sunt stocate astfel încât atunci când TIME = LENGH ele să poată fi reprezentate grafic pentru fiecare moment de timp din LENGH. La fiecare pas de simulare secvența de calcul executabilă este următoarea:

- Nivele;
- Auxiliare;
- Ritmuri.

Dacă valorile nivelelor sunt reținute pentru a fi reprezentate grafic, cele ale auxiliarelor și ritmurilor nu mai interesează în următorul pas de simulare, deci sunt șterse. De obicei după ce se efectuează simularea, valorile calculate ale tuturor nivelelor sunt reprezentate pe același grafic pentru a putea pune în evidență mai bine raporturile dintre comportamentele diferitelor variabile de nivel.

Apar astfel clar tendințele de ciclicitate corelate sau de anticiclicitate dintre aceste nivele.

În analizele efectuate se pot utiliza atât reprezentările grafice cât și valori tabelate, astfel încât să se poată remarca mai bine tendințele de evoluție încorporate în dinamica sistemului respectiv.

3.3.2. Limbaje de simulare utilizate în dinamica sistemelor

Pentru simularea numerică a modelelor de dinamică de sistem, pot fi utilizate atât limbaje de nivel înalt cum ar fi: C++, sau limbaje de simulare generalizate. Primul limbaj specializat a fost DYNAMO care a cunoscut două variante: DYNAMO1 și DYNAMO2. Ulterior au fost introduse și alte limbaje de simulare orientate către modelele de dinamică de sistem cum sunt: DYSMAP, VENSIM, STELLA, POWERSIM etc. Aceste limbaje au avut mai multe versiuni, astfel încât STELLA este astăzi la versiunea 9.2, iar POWERSIM STUDIO a ajuns la versiunea 7. Fiecare dintre acestea au fost tot mai performante, ajungându-se astăzi la limbaje de simulare extrem de performante care permit modelarea și simularea interactivă a sistemelor dinamice.

Un limbaj utilizat frecvent în domeniul tehnic este SIMULINK, un limbaj de simulare utilizat împreună cu MATLAB ce permite o mare flexibilitate în introducerea de funcții speciale, care sunt necesare pentru a descrie cât mai exact evoluțiile complexe ale sistemelor dinamice continue sau discrete.

Mai recent modelarea sistemelor dinamice, utilizând dinamica de sistem este posibilă și prin utilizarea unui limbaj de modelare bazat pe agenți, cum este NETLOGO. Dar despre acest tip de simulare a sistemelor economice dinamice vom vorbi în paragraful următor.

3.4. Simularea sistemelor bazate pe agenţi

Simularea modelelor cibernetice constituie o metodă larg utilizată în cibernetică pentru a studia şi a descoperi noi proprietăţi ale sistemelor adaptive complexe. Pe măsură ce concepţia şi metodele ciberneticii au evoluat, şi simularea s-a perfecţionat, ajutată evident şi de creşterea continuă a performanţelor calculatoarelor electronice. În prezent, aplicarea celor mai multe dintre modelele elaborate, fie că este vorba de modele bazate pe ecuaţii sau modele bazate pe agenţi, este precedată de simularea acestora, ceea ce dă posibilitatea evidenţierii caracteristicilor modelelor elaborate şi a performanţelor acestora încă înainte de a fi aplicate, fiind astfel posibilă corectarea şi îmbunătăţirea anumitor caracteristici sau proprietăţi.

Prima metodă de simulare aplicată în studiul sistemelor cibernetice este considerată Dinamica Industrială sau Dinamica Sistemelor, cum a fost ea denumită mai târziu, concepută de **Jay Forrester** şi frecvent aplicată în anii ’60-’70 ai secolului trecut. Datorită rolului ei în înţelegerea modului în care acţionează

buclele şi procesele feedback în sistemele economice, ne vom ocupa mai pe larg de această metodă în capitolul 6.

Apariţia şi dezvoltarea modelelor şi modelării bazate pe agenţi au reliefat rolul tot mai mare pe care simularea o are în analiza şi proiectarea sistemelor adaptive complexe. Au apărut astfel diferite limbaje de simulare, precum şi diverse metodologii prin care modelele bazate pe agenţi sunt simulate.

Cele mai cunoscute limbaje de modelare şi simulare din domeniul sistemelor multiagent sunt Swarm (Minar et al., 1996), Repast (Collier et al., 2003), JAS (Sonnessa, 2004), SPADES (Riley, 2003) şi Netlogo (Wilensky, 1999).

De regulă un limbaj de simulare interacţionează cu sistemele multiagent doar în faza de configurare. Acest lucru înseamnă, în esenţă, că după alegerea condiţiilor iniţiale ale sistemului complex, observatorul devine un spectator al evoluţiei simulate. Dacă estimarea variabilelor sistemului nu afectează în mod direct rezultatele simulării, acest lucru duce la un model de simulare corect. În alte cazuri, însă, metode alternative sunt necesare pentru a rezolva astfel de probleme. O astfel de metodă, numită simulare participativă (Resnick et al., 1998) prevede o modalitate prin care se extinde capacitatea de a interacţiona cu aceste sisteme în timpul simulării. Astfel, în cursul unei simulări participative, fiecare utilizator poate juca rolul unui sistem individual şi poate vedea cum comportamentul sistemului ca un întreg emerge din comportamentele individuale ale agenţilor. Astfel de medii virtuale promovează cooperarea, coordonarea şi negocierea între agenţii controlaţi de modele comportamentale prefixate (proiectate de utilizator) şi care fiind direcţionate de om pot urmări anumite scopuri. Comportamentul emergent al modelului şi relaţiile acestuia cu utilizatorul uman pot face dinamica sistemului simulate mult mai clară.

3.4.1. Principiile simulării bazate pe agenţi

Simularea şi lumea reală sunt două extreme ale unui spectru larg de sisteme, putând fi imaginată o gamă infinită de situaţii intermediare între acestea. Scopul principal al simulării este să modeleze un sistem real a cărui natură poate fi reprezentată prin aspecte concrete sau abstracte, fizice sau simbolice, discrete sau continue.

Pentru a obţine reprezentarea simulată a sistemului real trebuie totuşi să ne orientăm către domeniul computaţional discret. Acesta este distinct faţă de scopul sistemelor informaţionale obişnuite, care este să permită transmiterea informaţiei ce este deja formalizată în anumite organizaţii umane. Datorită acestui aspect, legat de trecerea de la un domeniu natural la unul sintetic, modelele de simulare sunt dezvoltate prin mai multe iteraţii, fiecare dintre aceste iteraţii producând un model îmbunătăţit al unui sistem, model ce este definit, implementat, verificat şi validat. Acest proces ciclic continuă până când modelul satisface obiectivele utilizatorului modelului respectiv.

Minsky spunea: „Pentru un observator B, un obiect A* este un model al unui obiect A în măsura în care B poate utiliza A* pentru a răspunde întrebărilor care îl interesează în legătură cu A" [MINSKY, 1965].

Deci, metoda simulării iterative diferă de metodele utilizate în cadrul sistemelor de procesare a informaţiei. Desigur, o simulare este şi ea un sistem sau produs software. Ideea este că, în timp ce în procesul de simulare, iteraţiile fac să evolueze de la un singur model iniţial către un model consistent util, iteraţiile ciclice în sistemele informaţionale inginereşti încearcă să atingă funcţii noi şi intercorelate ale sistemului sau software-ului respectiv. Acest lucru se realizează printr-un proces de construcţie modulară sau incrementală bazat pe cazuri sau experienţe anterioare. Cu alte cuvinte, în dezvoltarea sistemelor informaţionale, diferite funcţii care pot fi dezvoltate sunt bine cunoscute şi au o reprezentare apropiată de informaţia care trebuie să fie utilizată în organizaţia respectivă. În acest fel, sistemele informaţionale tind să provoace schimbări chiar în organizaţie. Din contră, atunci când se utilizează simularea în anumite aplicaţii, modelul de simulare furnizează rezultate destinate să ofere indicaţii privind sistemul real de care este interesat utilizatorul. Totuşi, utilizarea simulării nu induce sau precede o schimbare în sistemul real.

Succesul unei simulări este deci măsurat de cât de apropiat sau cât de fidel imită modelul de simulare sistemul real. Din această perspectivă, modelele de simulare bazate pe agenţi reuşesc să reproducă prin însăşi structura lor sistemele reale. Fiecare actor al unui proces sau sistem real poate să fie reprezentat în modelul bazat pe agenţi printr-un agent care simulează comportamentul actorului din sistemul real, independent de ceilalţi agenţi.

Una dintre caracteristicile esenţiale în aceste simulări este şi uşurinţa cu care se poate realiza validarea modelului respectiv. Utilizatorul poate pur şi simplu să exprime câteva idei scriind linii de cod într-un limbaj natural şi apoi începe simularea. În cursul evoluţiei el observă valorile unor variabile interesante predefinite şi poate lua decizii. Lucrări recente au permis reprezentarea pe display, în timp real, a rezultatelor simulării în două dimensiuni (RePast, JAS, Netlogo). În continuare se încearcă îmbunătăţirea acestei posibilităţi prin trecerea la reprezentarea tridimensională [Cacciaguerra, 2004].

Recent, ultimele versiuni ale lui Netlogo promovează un alt tip de vizualizare tridimensională [Wilensky, 2005].

Apar însă şi limite datorate posibilităţii apariţiei unor comportamente emergente complexe în orice simulare cu modele bazate pe agenţi. Două ar putea fi cauzele acestor limite. Prima cauză este legată de simplicitatea modelului bazat pe agenţi considerat. De regulă, modelele bazate pe agenţi sunt simple prin natura lor, ceea ce reflectă cunoştinţele actuale ale utilizatorului despre sistemul real modelat. Această simplitate se adaugă la reducţionismul cunoscut al oricărui tip de model. O astfel de limită este în măsură să afecteze acurateţea rezultatelor

obţinute, ştiut fiind faptul că modelul, cu cât este mai simplu, cu atât se îndepărtează de sistemul real pe care îl reprezintă.

De fapt, este şi foarte dificil să descriem exact toate comportamentele incluse într-un model datorită complexităţii intrinseci a interacţiunilor dintre actorii sociali. Deci, pentru a lăsa anumite grade de libertate, se introduc anumiţi paşi aleatori ceea ce determină însă o pierdere de precizie în analiză. În alte cazuri, nici nu este posibilă o descriere completă a comportamentelor agenţilor datorită legii nedeterminismului fizic al interacţiunilor dintre aceştia. Nu se pot descoperi toate interacţiunile dintre agenţi, unele fiind evidente, dar altele rămânând ascunse şi, din această poziţie, putând determina schimbări de comportament neaşteptate la agenţii între care există astfel de conexiuni ascunse.

Un al doilea motiv este legat de puterea de calcul mărginită. Software-ul obişnuit nu poate să prelucreze cantităţi mari de date într-o perioadă scurtă de timp datorită modalităţilor de prelucrare a acestora (secvenţială şi nu paralelă).În acest caz, atunci când suntem în faţa unor probleme tipice legate de simularea fizică, restricţia de timp nu poate fi depăşită în perioada scurtă de timp în care se fac simulările, deci experimentul cu modelele complexe devin imposibile. În plus, analiza unor sisteme fizice se poate realiza mai uşor decât a celor sociale datorită restricţiilor rigide şi necesităţi de a demonstra afirmaţiile care stau în spatele acestora.

Deci este destul de dificil de a implementa modele de simulare bazate pe agenţi care pot fi apoi folosite pentru a genera comportamente noi şi emergente. Totuşi, prin reducerea exigenţelor legate de precizia statistică a rezultatelor acestor modele, se pot depăşi, cel puţin în parte, dificultăţile descrise într-un mod eficient. Pentru aceasta este necesar ca modelele bazate pe agenţi să fie capabile să interacţioneze între ele rapid şi un timp suficient de lung în cadrul unui mediu virtual. În particular, sunt necesare următoarele:

- un protocol comun acceptat pentru schimbul de informaţii;
- un canal de comunicaţie cu banda largă;
- putere mare de calcul pentru a controla modelele de comportament.

Se crede, în general, că un mediu de joc cooperativ poate să satisfacă aceste cerinţe. Un joc cooperativ este un tip special de joc în care mai mulţi jucători joacă împreună pentru a atinge un scop anterior definit.

Simularea participativă realizată prin intermediul modelelor bazate pe agenţi constituie una dintre cele mai bune metode de implementare a jocurilor cooperative. Totuşi trebuie spus că jocul este un instrument şi nu scopul simulării. Una dintre cele mai importante avantaje prin trecerea de la abordarea ca pe un simplu program de simulare la abordarea prin jocuri cooperative a modelelor bazate pe agenţi este că, în cel de-al doilea caz, oamenii pot interacţiona direct cu agenţii din interiorul mediului virtual alăturându-se jocului. Deci, orice informaţie sau cunoştinţele precedente privind mijloacele de simulare fac metodologia de simulare bazată pe agenţi mai accesibilă. Deci devine posibilă utilizarea competenţei cunoştinţelor oamenilor pentru

a creşte acurateţea procesului de simulare iterativă. Metoda consideră că şi oamenii ce sunt implicaţi în procesul de simulare sunt sisteme care au un comportament complex, astfel că o creştere a complexităţii procesului se poate induce prin interacţiunea dintre model şi utilizator. În definirea obiectivelor jocului, metoda încurajează ca oamenii să aplice propriile lor modele sociale în adoptarea unor sarcini pe care să le rezolve prin interacţiunea cu modelul.

Acest lucru este similar procesul mintal prin care un utilizator scrie un model de simulare obişnuit cu ajutorul unui limbaj de simulare obişnuit. În plus, oamenii de regulă nu cer putere computaţională suplimentară atunci când interacţionează între ei în timp. Ei utilizează un limbaj comun pentru a realiza aceste interacţiuni. De fapt, în timp ce un limbaj de simulare oferă un protocol de schimb de informaţie între agenţi, un joc este înţeles de către oameni pornind de la regulile sale. Vizualizarea în spaţiul tridimensional eventual completată cu audio tridimensional) este cel mai rapid mod de a realiza interacţiunile dintre oameni. O astfel de modalitate corespunde naturii umane şi posibilităţii sale de a înţelege mediul înconjurător.

Deci, jocul cooperativ necesită doar crearea şi coordonarea mediului comun de schimb de informaţii (care reprezintă jocul). În acest mod, problema restricţiei de timp este de asemenea rezolvată. Astfel, jocul cooperativ arată că are şi alte proprietăţi interesante.

Totuşi, atunci când are loc rularea unui model de simulare, apar anumite întrebări legate de proiectarea experimentelor. Astfel de întrebări sunt următoarele:

1) Cum putem analiza comportamentul unui model comportamental într-un astfel de context?

2) Putem să presupunem că prin participarea unui număr mare de participanţi şi pe o perioadă lungă de timp simularea va determina obţinerea unor proprietăţi statistice favorabile?

3) Şi, presupunând că acest lucru este adevărat, cum putem găsi acest număr mare de oameni care este dispus să participe la simulare o perioadă lungă de timp?

3.4.2. Limbaje de simulare în modelarea multiagent

1) Swarm

Cel mai cunoscut limbaj de modelare şi simulare al sistemelor multiagent este Swarm, iniţiat de Chris Langton începând cu anul 1994 la Santa Fe Institute (http://www.santafe.edu). Prima versiune a fost disponibilă în 1996, în continuare dezvoltându-se diferite alte versiuni până în prezent.

Datorită modelelor de simulare dezvoltate de comunitatea Swarm, astăzi avem o idee mai clară asupra potenţialului şi limitelor modelelor de simulare bazate pe agenţi. Astfel, prin acest limbaj de simulare au fost abordate diferite probleme din teoria jocurilor (Dilema Prizonierului, Jocul minorităţii), biologie, epidemiologie, aplicaţii financiare, economie, geografie, apărare, industrie şi ştiinţe.

În esenţă, Swarm reprezintă o colecţie de biblioteci scrise în Objective-C care permit implementarea modelelor multiagent. Codul sursă Swarm este orientat obiect şi facilitează încorporarea obiectelor Swarm în programele de simulare. Aceste programe sunt ierarhice: pe primul nivel (numit şi observator swarm) se creează o interfaţă către nivelele inferioare. Aceste nivele (numite model swarm) implementează agenţii individuali, programează activitatea acestora, culege informaţie despre ei şi o schimbă pe baza cererilor observatorului swarm. Swarm conţine o mulţime de tutoriale care împart între ele linii de cod pentru a facilita astfel proiectarea unui model multiagent; de exemplu, managementul memoriei interne, menţinerea listelor sau programarea acţiunilor.

Grupul de Dezvoltare Swarm (http://www.swarm.org), extrem de activ din punct de vedere ştiinţific, a adus în acest domeniu trei mari contribuţii: 1) *O metodă simplificată de dezvoltare a modelelor de simulare bazate pe agenţi.* Simularea a cerut întotdeauna o metodologie şi metode care derivau din mijloace statistice, generatoare de numere aleatoare, reprezentarea prin plotere a evoluţiei în timp a sistemelor simulate etc. O bibliotecă cu astfel de instrumente simplifică mult elaborarea modelului, reduce timpul necesar programării pe calculator şi face costurile simulării acceptabile. 2) *O definiţie a unei scheme în proiectarea modelului.* Mult mai important însă este o definiţie a unei metodologii pentru scrierea modelelor. Grupul de Dezvoltare Swarm sugerează, de exemplu, să se păstreze o separare strictă între model (un program care simulează un sistem) şi observator (o mulţime de rutine care analizează modelul, culege date statistice şi le prezintă utilizatorului) Această metodă este mult mai elegantă şi prezintă o asemănare cu lumea reală, în care lucrurile se întâmplă şi cercetătorii le analizează din afară, fără să participe le evoluţia evenimentelor.

Separarea între model şi observatory poate avea o legătură cu distincţia ontologică dintre complexitatea proiectată şi complexitatea controlată, făcută de J. Casti (1986). Complexitatea proiectată exprimă ideea de complexitate percepută de sistemul însuşi, în timp ce complexitatea controlată este complexitatea observatorului percepută de sistemul însuşi. Cu alte cuvinte, complexitatea poate să nu fie o proprietate absolută a unui sistem ci să decurgă din interacţiunea dintre observatory şi sistemul observat. 3) *Crearea unei comunităţi de utilizatori.* Reunire unei comunităţi de oameni care utilizează Swarm şi care ţin legătura între ei prin e-mail şi un site web care colectează lucrările acestora reprezintă o contribuţie importantă a comunităţii.

Multe caracteristici utile nu sunt implementate în Swarm deşi, utilizând open source, biblioteci puternice pot fi introduse în simularea bazată pe agenţi. Totuşi, acest lucru duce la creşterea dificultăţii de a scrie şi a difuza modele bazate pe agenţi. De fapt, Swarm nu reprezintă doar o bibliotecă de simulare ci şi o metodologie, un cadru de referinţă pentru construirea modelelor în aşa fel încât

oricine, cunoscând interfaţa acestuia, poate înţelege uşor codul sursă şi verifica orice detaliu al modelului.

O primă caracteristică a lui Swarm este conceptul de *maşină virtuală*. Maşina virtuală permite descrierea comportamentelor agenţilor, unul câte unul, agent cu agent, context cu context, toate în condiţiile în care se culeg informaţii despre momentele de timp şi context. Swarm face, de asemenea, posibilă compunerea sau descompunerea ierarhiilor de agenţi. Această proprietate se numeşte *compozabilitate*. Această noţiune este utilă deoarece nu este întotdeauna clar unde începe un proces de modelare. De exemplu, în modelarea unei organizaţii mari, poate să apară cazul în care înţelegerea subiectivă a rolurilor şi responsabilităţilor indivizilor sau departamentelor diferă mult, ceea ce poate să conducă la performanţe slabe în unele situaţii şi performanţe bune în alte situaţii. În loc să se studieze cum lucrează organizaţia şi de ce apar astfel de diferenţe, se pot construi componente independente ale modelului din mai multe perspective şi apoi combina aceste componente (neglijând diferenţele privind percepţia indivizilor). O astfel de *metodă bottom-up* are avantajul documentării asupra diferenţelor în percepţia organizaţiilor determinate de senzitivităţile contextuale.

2) JAS (Java Agent-based Simulation)

JAS este: i) un mediu de simulare; ii) un cadru pentru construirea modelelor bazate pe agenţi; iii) o bibliotecă Java care conţine programe orientate către simulare.

JAS este un pachet de programe open source, găzduit de către portalul SourceForge şi constă dintr-o colecţie de utilitare Java compunând un cadru pentru construirea modelelor de simulare bazate pe agent. Biblioteca a fost dezvoltată pornind de la filozofia Swarm a cadrului model-observator.

Pentru ca modelele să fie într-adevăr standardizate, este necesar ca să se adauge mijloacelor de bază pentru realizarea modelelor o serie de alte caracteristice. Utilizarea bibliotecilor externe este bună, dar ele trebuie omogenizate cu caracteristicile mijlocului respectiv de modelare. JAS include biblioteci standard deja testate, dar ele apar ca fiind o parte a lui JAS pentru anumite clase specifice. De exemplu, ca plotter standard se utilizează biblioteca ptPlot7, dar interfaţa complexă a acesteia a fost filtrată şi acum este capabilă să reprezinte date statistice conţinute în pachetul *jas.stats*.

În acest fel, utilizatorul final nu trebuie să cunoască amănunte referitoare la implementarea lui JAS, aceasta fiind o problemă a dezvoltatorilor. Astfel, se poate utiliza extensiv codul open source ceea ce facilitează îmbunătăţirea continuă a pachetului.

În continuare, se enumeră câteva caracteristici ale lui JAS comparativ cu Swarm:

- o implementare pură codului Java, astfel încât este uşor de instalat şi configurat. Nu sunt utilizate biblioteci care să fie dependente de sistemul de operare;
- posibilitatea de a executa în paralel acţiunile agenţilor;
- un protocol de reţea (Sim2Web8) permite executarea simulărilor pe web şi interacţiunile utilizatorilor cu modelele de simulare prin intermediul unei pagini web;
- o mulţime redusă de instrucţiuni de deplasare a obiectelor adaptate după Starlogo9;
- compatibilitatea cu formatul XML.

Un generator puternic de numere aleatoare şi funcţii statistice luate din COLTlibrary 10:

- compatibilitatea cu biblioteci AI cum sunt GA, ANN şi CS, ceea ce permite implementarea inteligenţei agenţilor;
- simularea în timp discret, cu un emulator în timp real şi diferite reprezentări ale unităţii de timp (orar, zilnic, lunar, anual etc.);
- încărcarea dinamică a modelelor, ceea ce reduce problemele legate de configurarea variabilei CLASSPATH pentru executarea modelelor;
- Un protocol multi-run pentru executarea automatării parametrilor.

JAS nu este doar o bibliotecă ci şi o aplicaţie. După procedura de instalare, de fapt, ea poate fi pornită ca orice aplicaţie Java.

Un model JAS este chiar o aplicaţie Java bazată pe clase definite în pachetul JAS.jar, compilat cu un compilator Java standard (JDK, de exemplu). De aceea, după instalare, aplicaţiile pot fi startate ca orice aplicaţie Java.

Mai mult, este posibil să oprim o simulare şi să o repornim fără a întrerupe JAS.

Datorită acestor proprietăţi, putem să definim un protocol multi-execuţie JAS, util pentru automatizarea calibrării parametrilor.

Panelul de control (Figura 4.1) este principala fereastră prin care este posibil să:

- creăm un model specificând clasa, bibliotecile şi parametrii pentru motorul de simulare cu ajutorul unui editor de modele;
- încărca şi rula un model compilat (în format XML);
- edita un număr random, poziţia unei ferestre etc.
- deschide o fereastră de rezultate
- controla starea motorului de simulare (lista de evenimente, modelele rulate, ferestre etc.)

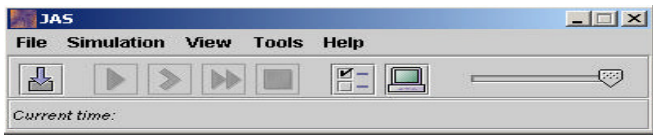


Figura 3.13. *Panelul de control JAS*

JAS este de fapt o clonă a lui Swarm obţinută prin translarea surselor Swarm scrise în Objective-C în Java. El conţine o bibliotecă java extinsă de obiecte pentru a modela, programa, reprezenta şi colecta date din simulile multi-agent. JAS permite reprezentarea vizuală a datelor obţinute din simulare prin intermediul histogramelor şi grafelor secvenţiale. Mai mult, el poate reprezenta evoluţia unui sistem complex simulat într-un format bidimensional (movie format).

3) NetLogo

NetLogo este un mediu de modelare programabil care permite utilizatorului final să dea instrucţiuni unui număr variabil de agenţi care operează în aceeaşi perioadă timp. El poate, de asemenea, implementa un instrument de simulare participativă (numit HubNet). HubNet conectează calculatoare în reţea din mediul NetLogo pentru a ajuta utilizatorul să controleze un agent în cursul unei simulări.

Ajuns la versiunea 5, NetLogo concurează cu Swarm şi RePast pentru cel mai utilizat mediu de modelare şi simulare multi-agent.