Proiectarea logică a sistemelor informatice

Dacă în primele etape, au fost identificate şi structurate cerinţele sistemului, în faza de proiectare logică se efectuează deplasarea atenţiei de la prezentarea a ceea ce există şi ce se intenţionează la descrierea a ceea ce va însemna noul sistem şi cum va funcţiona. Prezentarea noului sistem constă în prezentarea tuturor intrărilor sistemului, a ieşirilor, precum şi a interfeţelor şi dialogurilor.

1. Proiectarea formularelor/formatelor şi a rapoartelor

În cadrul etapei de analiză a sistemului informatic, intrările şi ieşirile au fost identificate şi prezentate, exprimând cerinţele informaţionale la nivelul fiecărui subsistem/ aplicaţie informatică. În acel moment nu s-au prezentat toate detaliile privind formularele/formatele, rapoartele şi procesul de modelare a datelor, insistându-se mai mult pe identificarea şi descrierea lor.

Fiecare format/formular de intrare va fi asociat unui flux al datelor de intrare într-un proces al DFD, iar rapoartele se pot regăsi într-un flux al datelor generate de un proces al DFD.

Un formular/format poate fi un document primar sau o machetă de ecran care conţine unele date predefinite, cărora li se adaugă altele ce urmează a fi completate în rubrici speciale.

Un raport este un document economic în care sunt incluse doar date predefinite, ceea ce înseamnă că poate fi numit şi document pasiv, folosit pentru a citi şi vizualiza informaţia.

În faza de proiectare logică se reprezintă doar o ciornă a formularelor/formatelor, rapoartelor sau ecranelor, ele fiind privite doar ca structură şi machetă, pot fi realizate cu ajutorul unui editor de texte sau un produs program orientat spre grafică sub forma unui prototip.

1.1. Proiectarea situaţiilor cu rezultate finale (rapoartelor)

Obiectivul prezentării detaliate a ieşirilor este şi acela de a determina formatul şi conţinutul tuturor rapoartelor imprimate şi ale documentelor şi ecranelor furnizate de sistem.

Proiectarea ieşirilor se va face astfel încât să servească pentru [11]:

- transmiterea rezultatelor prelucrării pe calculator utilizatorului, într-o formă pe care acesta să o înţeleagă şi în care să-şi regăsească cerinţele sale;
- transmiterea proiectului situaţiilor finale programatorului, fără ambiguităţi, pentru a-i permite acestuia trecerea la întocmirea programelor necesare editării sau vizualizării.

În proiectarea ieşirilor, analistul trebuie să elaboreze un model demonstrativ al raportului sau ecranului şi să întrebe utilizatorul dacă informaţiile din raport şi formatul acestuia sunt accesibile. Dacă ieşirile nu corespund cerinţelor utilizatorului, analistul trebuie să le modifice. Un

instrument util pentru formatul rapoartelor sau ecranelor realizate pe calculator îl constituie macheta imprimantei.

Macheta imprimantei este reprezentarea de detaliu a situaţiei de ieşire, cuprinzând:

- antet;
- titlu;
- date de identificare;
- cap de tabel;
- date elementare ce se imprima rând de rând;
- totalurile.
- detalii şi indicaţii tehnice de realizare care se referă la:
    - volumul datelor de ieşire;
    - frecvenţă;
    - număr de copii şi destinaţia fiecăreia;
    - grad de precizie al calculelor;
    - condiţii speciale de editare;
    - criterii de control, validare şi interpretare a datelor de ieşire.

Specificaţiile de ieşire vor cuprinde, pentru utilizator, macheta situaţiei iar pentru programator macheta situaţiei şi o serie de indicaţii tehnice de realizare.

Pe baza specificaţiilor de ieşire se trece la proiectarea fizică prin care se alege suportul informaţiilor de ieşire, se realizează definitivarea formei şi formatului de editare a situaţiilor (aşezarea în pagina / ecran, spaţierea între coloane şi rânduri, etc.) şi se definitivează procedurile de utilizare şi interpretare a ieşirilor.

Alegerea tipului de suport fizic de ieşire (imprimanta, display, disc fix, floppy disc, banda magnetică etc.) se face în funcţie de: timpul de răspuns solicitat, amplasarea utilizatorului faţă de calculator, hard-ul şi soft-ul existent, costul suportului, măsura în care răspunde necesităţilor de redare a conţinutului informaţional al situaţiei finale.

Când se pregătesc ieşirile, este bine să se ia în calcul ce se urmăreşte prin ele, astfel încât apelarea la categoriile de date de mai sus să se efectueze în cunoştinţă de cauză.

În definitivarea formei şi formatului de prezentare a situaţiilor finale trebuie să ţinem seama de o serie de considerente practice cum ar fi [11]:

- Respectarea unor cerinţe ale factorilor de decizie privind macheta situaţiei finale;
- Restricţii tehnice;
- Elemente de eficienţă;
- Lizibilitate – spaţiere;

- Utilizarea formularelor pretipărite;
- Utilizarea monitoarelor;
- Utilizarea generatoarelor de rapoarte.

Respectarea unor cerinţe ale factorilor de decizie privind macheta situaţiei finale

O serie de cerinţe ale conducerii privind macheta situaţiei finale obligă proiectantul la o anumită structurare şi machetare a situaţiilor finale. Informaţiile se pot împărţii în două grupe prin prisma sistemelor informatice interne şi externe. Informaţiile interne reprezintă acele informaţii culese, generate sau folosite în interiorul organizaţiei. Informaţiile externe se referă la cele colectate sau create de la sau pentru parteneri străini (facturi, rapoarte anuale, etc).

În funcţie de informaţiile care pot fi văzute din punct de vedere al echipei manageriale distingem: informaţii curente, de atenţionare, indicatori de bază, etc.

Restricţii tehnice

În proiectarea situaţiilor finale intervin o serie de restricţii datorate caracteristicilor şi performanţelor tehnice ale echipamentelor periferice şi anume: numărul maxim de caractere pe linie; numărul maxim de linii pe pagina / ecran; facilităţile de imprimare etc. Pe piaţă se afla o gamă variată de echipamente de redare a rezultatelor. Există mai multe tipuri de imprimante și console, ceea ce creează posibilitatea unei alegeri adecvate a perifericelor destinate obţinerii diverselor tipuri de situaţii finale.

Elemente de eficienţă

În proiectarea situaţiilor finale nu trebuie sa scape atenţiei şi aspectele de eficientă economică privind: reducerea timpului calculator consumat cu editarea propriu-zisa a situaţiilor; economie de hârtie de imprimantă. Abilitatea şi experienţa proprie a programatorilor joacă în acest sens un rol important.

În vederea optimizării obţinerii situaţiilor finale pe imprimantă se pot folosi de la caz la caz, diverse tehnici cum ar fi: editarea mai multor tabele pe aceeaşi pagină de imprimantă; editarea unei situaţii imprimând faţă/verso pe aceeaşi coală;

Pentru a nu irosi timp cu editarea unor situaţii finale voluminoase se recomandă mai întâi rularea unor programe scurte care să verifice cheile de control aplicate. Numai dacă aceste chei sunt corecte, eventual verificate şi de utilizator, se poate lansa editarea analitica a situaţiilor finale. Programele care editează situaţii finale voluminoase trebuie prevăzute cu posibilitatea de întrerupere (respectiv de reluare a editării în cazul unor incidente ivite în timpul rulării) sau editarea

lor sub forma unui fişier ASCII sau text pe hard disc sau floppy disc, urmând imprimarea ulterioara a acestui fişier, total sau parţial.

Lizibilitate – spaţiere

Parcurgerea unei situaţii finale trebuie să fie cât mai uşoara, "citirea" unei situaţii nu trebuie să dea naştere la ambiguităţi. Este necesar ca situaţia sa fie autoexplicativă. Pentru aceasta, antetul va conţine informaţii şi coduri ce vor indica sursa de emitere a raportului, exprimând clar, sintetic, conţinutul raportului şi perioada la care se referă.

Capul de tabel, împreuna cu titlul şi antetul, se afişează pe următoarele pagini numai dacă au intervenit schimbări în cadrul caracteristicilor de grupare faţă de prima pagină, altfel se imprimă doar numerotarea coloanelor de tabel.

Informaţiile importante pot fi subliniate. Totalurile se separă de informaţiile analitice. Informaţiile care se repetă pe linii succesive se imprimă o singură dată.

Utilizarea formularelor pretipărite

Aceasta implica utilizarea unei hârtii de imprimanta ce cuprinde elemente fixe ale situaţiei finale, cum ar fi antetul, titlul, capul de tabel, textul explicativ etc. Aceasta conduce la o creştere a vitezei de editare şi o diminuare a uzurii imprimantelor. Totodată situaţiile obţinute sunt mai estetice şi sunt uşor de parcurs de utilizatori.

Utilizarea generatoarelor de rapoarte (REPORT WRITER)

Multe limbaje de programare, pachete de programe şi sisteme de gestiune a bazelor de date dispun de module specializate în editarea de rapoarte, ceea ce conduce la reducerea considerabila a eforturilor programatorilor. De obicei, aceste generatoare solicită precizarea titlului, antetului de coloană, conţinutul unui rând de date (de detaliu), gradele de total şi maniera lor de afişare, la începutul sau la sfârşitul grupului de date, al paginii sau raportului. De asemenea, se pot selecta dimensiunea unei linii, coloane, pagini, spaţierea dintre linii, coloane, afişarea datelor privind momentul listării, statistici etc. Astfel de module specializate există în pachete de programe pentru gestionarea bazelor de date cum ar fi: ACCESS, d'BASE, ORACLE, FOXPRO, PARADOX, etc.

1.2. Proiectarea codurilor

În proiectarea sistemului de coduri trebuie să avem în vedere două aspecte importante şi anume [11]:

- influenţa tipului şi structurii codului asupra performanţelor sistemului informatic;

– implicaţiile utilizării codurilor în operaţiile de culegere a datelor şi interpretarea rezultatelor finale de către utilizatorii neinformaticieni.

Primul aspect ridică probleme de ordin tehnic în realizarea nomenclatorului de coduri şi are în vedere facilitarea operaţiilor de prelucrare, ocuparea unui spaţiu de memorie internă şi externă cât mai mic etc.

Celui de-al doilea aspect trebuie să i se acorde o atenţie mai mare în vederea uşurării activităţilor de culegere, verificare a datelor şi interpretarea rezultatelor din situaţiile finale. Având în vedere aceste considerente, se impune ca la proiectarea unui sistem de coduri să se respecte o serie de cerinţe.

Activităţile parcurse în realizarea unui sistem de coduri sunt:

– analiza elementelor ce urmează a fi codificate;
– precizarea şi uniformizarea tehnologiei, a denumirilor;
– stabilirea caracteristicilor şi a relaţiilor dintre elementele de codificat;
– alegerea tipurilor de coduri; estimarea capacităţii, lungimii şi formatului codurilor;
– atribuirea codurilor elementelor de codificat (crearea nomenclatoarelor de coduri);
– întreţinerea nomenclatoarelor de coduri.

Este indicat a se utiliza, acolo unde este cazul, sistemele de codificare existente la nivelul economiei naţionale (CAEN, SIRUES, SIRUTA, CNP, etc).

1.3. Proiectarea intrărilor în sistemul informatic

Datele de intrare parcurg o succesiune de etape până la utilizarea efectivă în cadrul sistemului informatic. Aceste etape intermediare sunt: înregistrarea datelor pe documentul de intrare; conversia datelor într-o formă acceptata de sistemul de calcul / transpunere pe suportul tehnic; verificarea sintactică şi semantică a datelor de intrare; corecţia datelor eronate etc.

La proiectarea intrărilor este necesar să se realizeze, în principal următoarele activităţi:

- alegerea suportului tehnic pentru culegerea datelor;
- proiectarea machetelor documentelor de intrare, stabilirea instrucţiunilor de culegere;
- stabilirea regulilor de control şi de validare a datelor;
- proiectarea formularelor (videoformatului) de intrare.

Alegerea suportului tehnic al datelor de intrare se face în funcţie de cerinţele aplicaţiei informatice. Datele introduse de la terminal, fie intră imediat în circuitul de prelucrare-actualizare a unei baze de date, fie sunt stocate pe un suport magnetic sau sunt stocate în vederea prelucrării ulterioare.

La proiectarea machetei documentelor de intrare (indiferent de metodele de prelucrare a datelor folosite ulterior) sunt respectate câteva reguli care să înlesnească completarea şi apoi utilizarea documentului atât pentru prelucrarea automată a datelor cât mai ales pentru necesităţile curente ale salariaţilor societăţii economice. Nu este recomandabil să dublăm documentele primare, prin proiectarea unor documente destinate exclusiv preluării datelor pentru necesităţile prelucrării automate.

Macheta documentului primar trebuie să conţină:

- antetul–ce reprezintă denumirea unităţii şi/sau a serviciului care-l emite;
- denumirea documentului;
- coduri de identificare,
- data documentului;
- rubrici /casete/ rânduri pentru denumirea informaţiilor cantitativ-valorice şi coduri;
- rubrici /casete /spaţii pentru semnături şi ştampile;
- text explicativ, eventual indicaţii de completare şi verificare.

În ordonarea informaţiilor pe document, deci în rubricarea documentului se va ţine seama de câteva reguli: antetul se plasează în stânga sus; codurile şi alte informaţii de identificare se plasează în dreapta sus; sensul natural de parcurgere este de sus în jos şi de la stânga la dreapta; zonele de document ce se completează de compartimente/ persoane diferite se marchează / grupează distinct; mărimea şi spaţierea documentului, distanţa dintre rânduri, dimensiunea rubricilor, depind de locul şi modalitatea de completare (manual, dactilo, automat) precum şi de nivelul de calificare a personalului ce completează documentul.

Aşezarea rubricilor pe document trebuie să respecte ordinea firească de folosire a documentului şi nu ordinea de utilizare a datelor în programe. Ordinea de culegere a datelor este suficient a fi precizată prin numerotarea rubricilor sau simpla lor încadrare în chenar sau utilizarea de litere îngroşate în denumirea rubricilor implicate în dialogul operator-calculator.

Atunci când documentul există într-o formă pe hârtie, în varianta pe calculator se va urmări păstrarea în mare măsură a structurii existente, dar cu adaptări specifice noului mediu.

Regulile de control şi procedurile de validare a datelor de intrare trebuie să cuprindă [11]:

- reguli de verificare a volumului, secvenţei documentelor şi a cifrelor de control (dacă este cazul) pe pachetele de documente primare;
- reguli pentru controlul sintactic şi semantic a datelor din documentele primare. Aceste reguli se referă la: încadrarea indicatorilor numerici (în limitele de verosimilitate), corelaţii logice (între indicatorii înscrişi în acelaşi document, sau cu alţi indicatori din baza de date), prezenţa unor informaţii obligatorii (apartenenţa codurilor la

nomenclatoarele de coduri specifice aplicaţiei informatice) etc. Aceste reguli sunt necesare pentru scrierea programelor de verificare logică a datelor de intrare.

Proiectarea formularelor(videoformatelor) **de intrare** pentru introducerea datelor de intrare se face în funcţie de modul concret de desfăşurare a dialogului operator-calculator. Acest dialog se poate desfăşura sub următoarele forme:

- întrebare-răspuns cu defilarea liniilor ecranului;
- afişare pe ecran a machetei de introducere a datelor de intrare.

În prima variantă, mai uşor de realizat practic, operatorul introduce succesiv, ca răspuns la întrebările afişate pe ecran, date de intrare. La proiectarea formelor de intrare este necesar ca proiectantul să aibă în vedere o serie de aspecte cum ar fi:

- afişarea la un moment dat a unui volum redus de informaţii;
- afişarea la un moment dat a unei cereri de răspuns ce se referă la o singură informaţie;
- scurtarea răspunsului operatorului prin folosirea mnemonicelor şi codificărilor;
- utilizarea unor formate clare şi precise pentru afişare;
- evitarea cuvintelor şi caracterelor dificil de citit sau înţeles;
- existenta unor rutine speciale de tipul HELP;
- preluarea asistată a unor coduri (ex. utilizare tehnici de tip *Lookup wizard* în ACCESS)

În varianta a doua cursorul marchează de fiecare dată câmpul curent care se introduce. Ecranul display-ului trebuie să reproducă integral sau simplificat macheta documentului, respectând aceeaşi ordine a rubricilor. Mesajele de eroare se pot afişa într-o zona prestabilita a ecranului, însoţită de avertizare sonora sau luminoasa.

Dacă este cazul, pentru unele câmpuri (rubrici) de date se pot prelua valori implicite, atunci când nu sunt completate. Aceste valori se preiau din nomenclatorul de coduri, fişierele bazei de date sau tabele special memorate pentru valorile asumate prin lipsa sau prin aplicarea unui algoritm. Pentru o mai uşoară operare este necesar să folosim toate facilităţile de afişare şi de combinare a culorilor (figura 1).

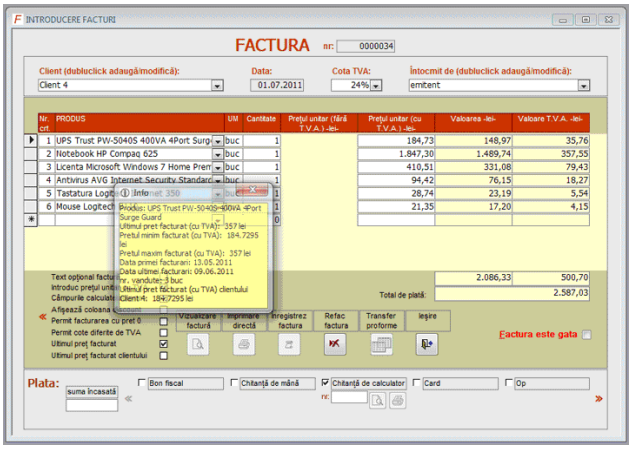

Figura 1. Posibil ecran al unei aplicaţii de facturare

În proiectarea formularelor de intrare pot fi utilizate componente specializate în acest sens din sistemele de gestiune a bazelor de date cum ar fi ACCESS, dBASE, ORACLE, PARADOX precum şi programe scrise în diverse limbaje de programare.

2. Proiectarea interfeţelor şi a dialogurilor

Interfaţa cu utilizatorul reprezintă o parte a aplicaţiei software care permite utilizatorilor să-si exprime intenţiile de operare asupra calculatorului şi să interpreteze rezultatele acţiunilor efectuate de maşină. Prin proiectarea dialogurilor şi a interfeţelor se definesc modalităţile prin care utilizatorii şi calculatoarele schimbă informaţii [46].

Metode şi echipamente folosite în dialogul om-maşină

Interfaţa om – maşină defineşte modalitatea prin care utilizatorul interacţionează cu un sistem informatic. Interfeţele sunt destul de variate, conform descrierilor, însă toate trebuie să dispună de un stil sau de o metodă prin care să se folosească anumite echipamente în măsură să faciliteze o astfel de interacţiune.

Metode de interacţiune

Metodele cele mai utilizate sunt:

- interacţiunea prin limbaj-comandă (în acest tip de interacţiune utilizatorul transmite calculatorului comenzile sub forma unui şir de caractere);
- interacţiunea prin meniuri(utilizatorul transmite comenzile sale calculatorului prin intermediul unui sistem de meniuri şi opţiuni de meniu sau folosind scurtături sub formă de combinaţii de taste);
- interacţiunea bazată pe obiecte *icons*(comunicarea se face prin intermediul pictogramelor. La apăsarea lor se activează o anumită funcţie sau comandă);
- interacţiunea prin limbaj natural(comenzile se transmit folosind vocea şi sintetizatoarele de voce pentru redarea rezultatelor).

Echipamentelor necesare interacţiunii cu sistemul

Cele mai folosite echipamente sunt:

- *keyboard* – tastatura este formată dintr-un set de butoane (taste) Prin intermediul ei se introduc date, comenzi;
- *mouse;*
- *joystick;*

- *touch screen* – atingerea ecranului constituie modalitatea prin care are loc selecţia;
- *light pen* – stiloul optic, efectuează selecţia prin apăsarea pe ecran;
- *voice* – vocea constituie modalitatea de transmitere a textelor şi comenzilor către calculator.

Proiectarea dialogurilor

Proiectarea dialogurilor este procesul prin care sunt proiectate toate secvenţele folosite de utilizator pentru a comunica cu un sistem informatic. Rolul proiectantului este de a selecta cele mai potrivite metode şi echipamente, precum şi de a prezenta condiţiile în care se pot afişa informaţiile sau se pot obţine de la utilizator.

Pentru a obţine rezultate bune trebuie să se ţină seama de regulile de bază la conceperea dialogurilor cum ar fi: uniformitate, comenzi scurte, uşurinţa în lucru, controlul, operaţiunea inversă (refacerea unui element şters), rezolvarea erorilor, etc.

O modalitate de prezentare a secvenţei dialogurilor este cea care apelează la tehnica diagramelor prin care se reprezintă meniurile componente ale aplicaţiei.

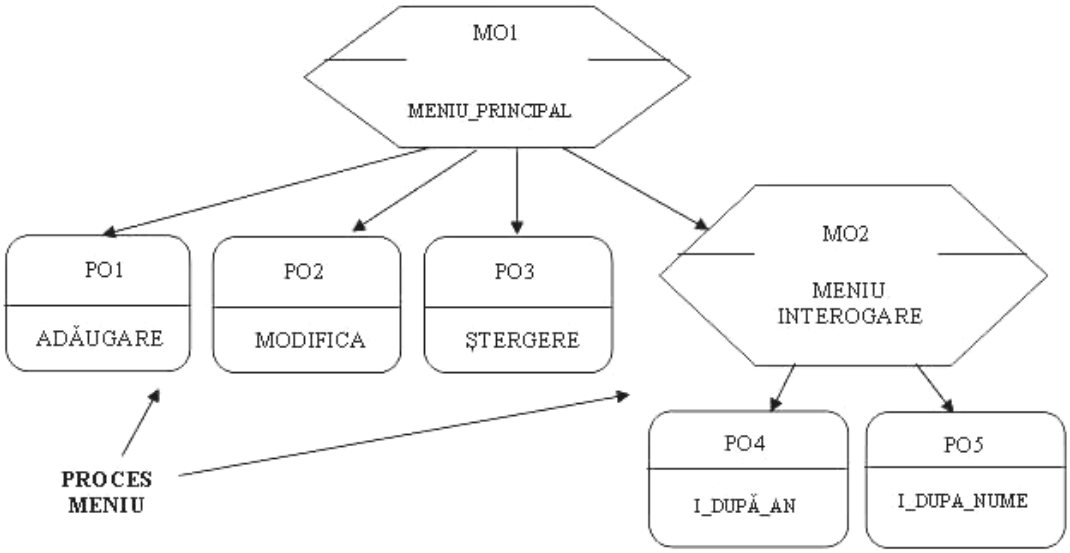

Fig. 4.2. Exemplu de diagramă de apelare a meniurilor.

Pentru proiectarea interfeţelor şi dialogurilor se poate apela la ajutorul oferit de produsele CASE sau la mediile de dezvoltare grafică ACCESS, Visual Basic, etc. Pentru a se putea proiecta în condiţii optime mediile GUI (*Graphical User Interface*) trebuie să se cunoască aceste medii. În mediile grafice informaţiile se plasează în ferestre. Acestea au trăsături specifice ca: redimensionarea, maximizarea, disponibilitatea la deplasare, meniu sistem, etc.

3. Proiectarea logică a bazelor de date

Proiectarea logică a bazelor de date este în strânsă legătură cu modelarea conceptuală a datelor, aceasta însemnând reprezentarea modului de organizare a datelor, independent de tehnologiile specifice de prelucrare a bazelor de date. Procesul de modelare logică a datelor se derulează în paralel cu celelalte activităţi ale proiectării logice: **proiectarea formularelor** şi a **rapoartelor** şi **proiectarea dialogurilor** şi **interfeţelor**. Modelarea logică a datelor se realizează nu numai pe baza diagramei entitate-relaţie, ci şi pe baza machetelor formularelor şi a rapoartelor. Se efectuează analiza datelor elementare din intrările şi ieşirile sistemului pentru a se desprinde legăturile dintre ele.

Prin modelarea logică a datelor se urmăreşte:

- structurarea performantă a datelor prin procesul de normalizare;
- obţinerea unui model logic al datelor din care să se poată realiza proiectul bazei de date fizice funcţie de tipul de SGBD utilizat: relaţional – cel mai utilizat în prezent, reţea, ierarhic, sau orientate-obiect;
- realizarea unui model al datelor care să răspundă cerinţelor actuale de date regăsite în formulare şi rapoarte. Modelarea logică este un proces ascendent (*bottom-up*, de jos în sus) de obţinere a relaţiilor din formulare şi rapoarte prin transformarea modelului entitate-relaţie într-o formă relaţională.

Modelarea logică a datelor descrie datele cu ajutorul unei notaţii speciale, care corespunde unui mod de organizare a acestora de către un sistem de gestiune a bazelor de date. Procesul de modelare a datelor este complex. În fiecare etapă a ciclului de viaţă se regăseşte câte o activitate specifică datelor după cum urmează [46]:

- Analiza – Modelele conceptuale ale datelor, prezentarea DER;
- Proiectarea logică – Modelul logic al datelor (relaţional);
- Proiectarea fizică – Proiectarea fizică a bazelor de date şi a fişierelor (organizarea fişierelor);
- Implementarea – Descrierea fişierelor şi a bazelor de date.

După cum prezintă profesorul Oprea D. În "Analiza şi proiectarea sistemelor informaţionale economice" în procesul de modelare logică există patru paşi esenţiali:

1. Realizarea unui model logic al datelor din perspectiva utilizatorului (formulare şi rapoarte) privind aplicaţia, folosindu-se principiile normalizării;
2. Contopirea tuturor perspectivelor normalizate ale utilizatorilor într-un model logic consolidat (centralizat) al datelor, numit şi integrarea perspectivelor;
3. Transformarea modelului conceptual al datelor (entitate-relaţie), realizat fără să se ţină cont de perspectiva utilizatorului, într-un set de relaţii normalizate;

4. Compararea modelului logic consolidat al datelor cu modelul transformat al entităţii-relaţie şi realizarea, prin integrarea perspectivelor, a unui model logic final al datelor aplicaţiei.

Rezultatele obţinute prin parcurgerea celor patru paşi pot fi ilustrate prin intermediul unor exemple oferite în literatura de specialitate de McFadden şi Hoffer.

Primul pas al modelării logice poate fi ilustrat prin două rapoarte solicitate de utilizatori, reprezentând perspectiva utilităţii sistemului din punctul lor de vedere:

- cel mai bun client al produsului X;
- situaţia comenzilor în curs.

Ecranul "Cel mai bun client al produsului X", prin percepţia utilizatorului, are următorul format:

```
Cel mai bun client al produsului
Introduceţi codul produsului:      P1122
Data de început:                   1/01/2016
Data de sfârşit:                   31/03/2016
- - - - - - - - - - - - - - - - - - - - - - - - - - - -
COD CLIENT:                        1111
NUME CLIENT:                S.C. ALPHA S.R.L.
VOLUM:                      1000
```

Fig. 4.3 Model de ecran solicitat de utilizatori [46]

Din analiza ecranului se pot desprinde relaţiile:

CLIENT(<u>COD_CLIENT</u>, NUME)
COMANDA(<u>NR_COMANDA</u>, <u>COD_CLIENT</u>, DATA_COMANDA)
PRODUS(<u>COD_PRODUS</u>)
COMANDA(<u>NR_COMANDA</u>,<u>COD_PRODUS</u>,CANTITATE_COMANDATA)

```
                                                    Pagina 1
Situaţia comenzilor în curs
31/03/2016

COD PRODUS
        CANTITĂŢI_DE_LIVRAT
A1111                       0
A2222                       0
B1111                     150
Y9999                     100
```

Fig. 4.4. Model de raport solicitat de utilizatori [46]

Realizarea raportului este posibilă prin folosirea următoarelor entităţi:

PRODUS(COD_PRODUS), COMANDA(NR_COMANDA, DATA_COMANDA)
COMANDA(NR_COMANDA, COD_PRODUS, CANTITATE_COMANDATA)
LIVRARE(COD_PRODUS, NR_FACTURA, CANTITATE_LIVRATA)
FACTURA(NR_FACTURA, DATA_FACTURA)

Pasul al doilea presupune comasarea perspectivelor utilizatorilor şi crearea unui set integrat al relaţiilor, rezultând următoarele relaţii (tabele):

CLIENT(COD_CLIENT, NUME)
PRODUS(COD_PRODUS)
FACTURA(NR_FACTURA, DATA_FACTURA)
COMANDA(NR_COMANDA, COD_CLIENT, DATA_COMANDA)
LINIE_COMANDA(NR_COMANDA, COD_PRODUS, CANTITATE_COMANDATA)
LIVRARE(COD_PRODUS, NR_FACTURA, CANTITATE_LIVRATA)

Pasul al treilea constă în transformarea modelului conceptual al datelor (diagrama entitate-relaţie) din aplicaţie fără să se ţină cont de punctul de vedere al utilizatorului, într-un set de relaţii normalizate. Din analiza diagramei din figura 4.5 se desprind următoarele relaţii:
CLIENT(COD_CLIENT, NUME, ADRESA)
PRODUS(COD_PRODUS, DENUMIRE)
FACTURA(NR_FACTURA, NR_COMANDA)
COMANDA(NR_COMANDA, COD_CLIENT)
LINIE_COMANDA(NR_COMANDA, COD_PRODUS, CANTITATE_COMANDATA)
LIVRARE(NR_FACTURA, COD_PRODUS, CANTITATE_LIVRATA)

Pasul al patrulea compară modelul obţinut din pasul doi cu cel din pasul trei şi integrează perspectivele utilizatorilor în vederea obţinerii unui model logic final, după cum urmează:

CLIENT(COD_CLIENT, NUME, ADRESA)
PRODUS(COD_PRODUS, DENUMIRE)
FACTURA(NR_FACTURA, NR_COMANDA, DATA_FACTURA)
COMANDA(NR_COMANDA, COD_CLIENT, DATA_COMANDA)
LINIE_COMANDA(NR_COMANDA, COD_PRODUS, CANTITATE_COMANDATA)
LIVRARE(NR_FACTURA, COD_PRODUS, CANTITATE_LIVRATA)

Rezultatul modelării logice a datelor îl constituie relaţiile normalizate rezultate din cel de-al patrulea pas al procesului. De asemenea, alt rezultat se va concretiza în actualizarea depozitului (*repository*) sau a dicţionarului proiectului. **Diferenţa majoră între modelarea conceptuală şi cea logică este că după modelarea logică a datelor cerinţele structurate de date se concretizează în relaţii, şi nu în entităţi.** Din cauza normalizării nu este necesară o corespondenţă unu-la-unu între entităţi şi relaţii.

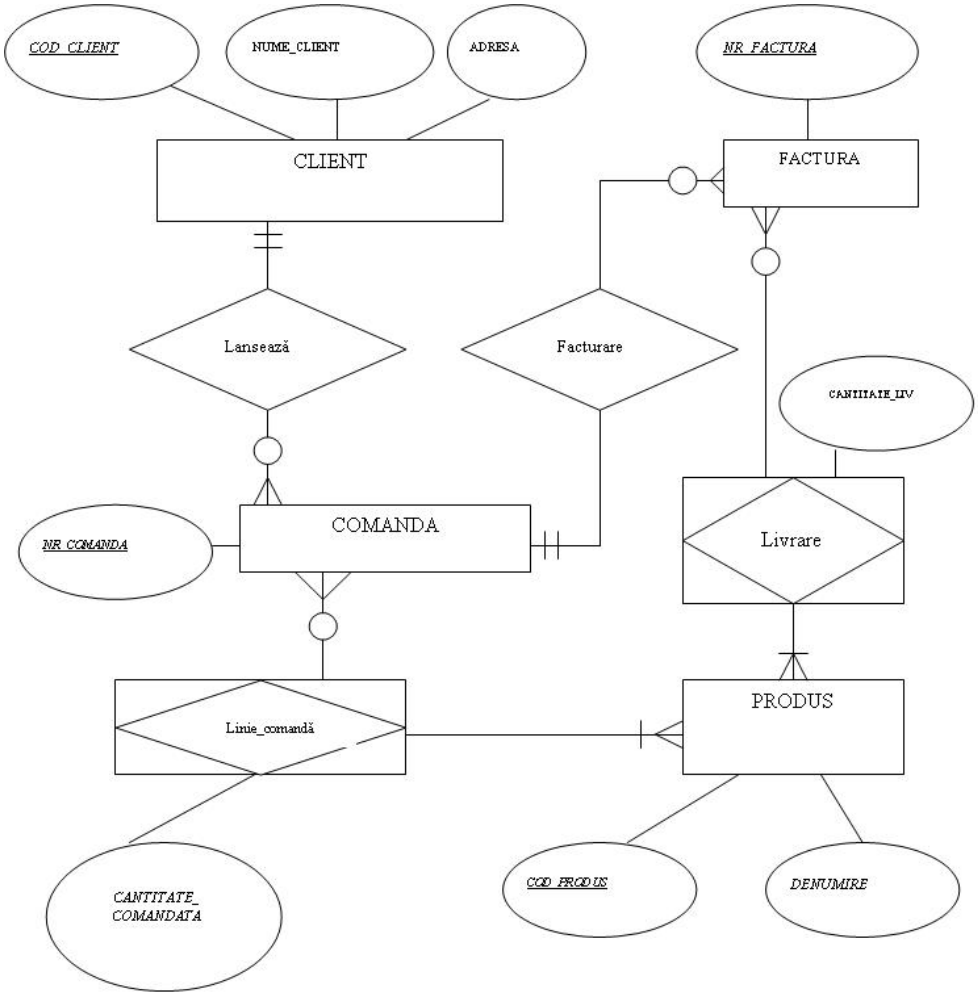

Fig. 4.5. Diagrama entitate-relaţie pentru gestiunea clienţilor [46]

3.1. Normalizarea relaţiilor - Forme normale

Între atributele unei relaţii sau între atribute din relaţii diferite pot exista anumite legături logice (dependenţe), care influenţează proprietăţile schemelor de relaţie în raport cu operaţiile curente care intervin în timpul exploatării bazei de date: adăugare, ştergere, actualizare. Aceste legături logice, cunoscute în literatura de specialitate sub denumirile de dependenţele funcţionale, dependenţele multivalorice şi dependenţe de cuplare, au implicaţii majore asupra criteriilor de proiectare a schemelor relaţionale. Alegerea unui model conceptual convenabil pentru o bază de date relaţională poate necesita realizarea unor descompuneri pentru anumite scheme de relaţie date,

descompuneri care să izoleze dependenţele existente şi prin aceasta să elimine anomaliile care se datorează acestor dependenţe.

Dependenţe funcţionale

Pentru definirea acestui tip de dependenţe se consideră schema de relaţie

Prestari_Servicii (Cod, Nume_client, Adresa, Serviciu_prestat, Valoare)

definită pentru urmărirea serviciilor prestate de o firmă pentru diverşi clienţi.

Atributul *Adresa* este dependent de atributul *Nume_client* (presupunând că fiecare client are o singură adresă, rezultă că fiecare valoare a atributului *Nume_client* determină în mod unic valoarea corespunzătoare a atributului *Adresa).* Analizând schema de relaţie de mai sus, se constată că atributul *Adresa* este redundant, deoarece valoarea acestuia este repetată pentru fiecare serviciu prestat pentru acest client, ceea ce conduce la apariţia următoarelor anomalii:

– Anomalia la adăugare:

adresa unui client poate fi preluată numai după ce pentru acesta se prestează cel puţin un serviciu.

– Anomalia la ştergere:

dacă se şterg toate serviciile prestate pentru un anumit client, se pierde adresa acestui client.

– Anomalia la actualizare:

datorită redundanţei relativ la atributul Adresa, dacă se schimbă adresa unui client este necesară parcurgerea întregii relaţii pentru a identifica şi actualiza toate apariţiile acestui client, în caz contrar baza de date devine inconsistentă (acelaşi client poate apare la adrese diferite).

Aceste anomalii pot fi eliminate, dacă schema de relaţie *Prestari_Servicii* se înlocuieşte prin următoarele două scheme de relaţie:

Clienti(Cod, Nume_client, Adresa)

Servicii(Cod, Serviciu_prestat, Valoare).

Relaţia *Clienti* conţine codul, numele şi adresa fiecărui client, fără nici un fel de redundanţă, iar relaţia *Servicii* conţine serviciile prestate pentru fiecare client şi valorile acestor servicii. Un dezavantaj al descompunerii relaţiei iniţiale în cele două relaţii este acela că pentru a determina adresa clientului pentru care s-a prestat un anumit serviciu este necesară efectuarea unei operaţii de cuplare a relaţiilor *Clienti* şi *Servicii*.

Se consideră o schemă de relaţie R şi A,B două atribute simple sau compuse ale schemei de relaţie R. Atributul A determină funcţional atributul B sau B depinde funcţional de A, dacă şi numai dacă oricărei valori a atributului A îi corespunde o singură valoare a atributului B (se notează A->B).

Dependenţa funcţională A->B este totală dacă nu există nici un subset C al atributului A ($C \subset A$) astfel încât C->B şi este parţială în caz contrar.

În relaţia *Prestari_Servicii*, una din dependenţele funcţionale care poate fi pusă în evidenţă este *Nume_client->Adresa.*

Deoarece într-o relaţie orice cheie identifică în mod unic fiecare tuplu al relaţiei, deci determină în mod univoc valorile atributelor tuplului, rezultă că în orice relaţie atributele sunt dependente funcţional faţă de cheile acesteia.

Se pot face, până în acest moment, următoarele precizări:

Eliminarea dependenţelor funcţionale din schemele de relaţie şi a consecinţelor negative (redundanţa datelor; anomaliile de adăugare, ştergere, actualizare) se realizează prin descompunerea schemei date într-o colecţie de scheme mai simple în care sunt evitate neajunsurile mai sus menţionate. Reconstituirea relaţiei iniţiale se poate face prin operaţia de cuplare (uniune). Pentru ca descompunerea schemei de relaţie să fie echivalentă cu relaţia iniţială, trebuie să fie îndeplinite condiţiile:

- cuplare fără pierdere de informaţie;
- conservarea dependenţelor (dependenţele funcţionale din relaţia iniţială trebuie să se regăsească în relaţiile rezultate prin descompunere).

Formele normale sunt scheme de relaţie echivalente obţinute prin descompunerea unor scheme de relaţie în vederea eliminării redundanţei datelor şi anomaliilor la adăugare, actualizare, ştergere înregistrări în baza de date. Descompunerile schemelor de relaţii în scheme de relaţii echivalente având în vedere dependenţele funcţionale conduc la definirea primelor 4 nivele de forme normale şi anume: prima formă normală (FN1), a doua formă normală (FN2), a treia formă normală (FN3) şi forma normală Boyce-Codd (FNBC).

A patra formă normală (FN4) este definită având în vedere dependenţele multivalorice, iar a cincea formă normală (FN5) este definită având în vedere dependenţele de cuplare. Începând de la prima formă normală şi până la forma normală FN5 se impun condiţii din ce în ce mai restrictive asupra relaţiilor. Astfel o relaţie aflată pe un anumit nivel de normalizare (FN5) satisface toate restricţiile cerute de nivele inferioare de normalizare (FN1, FN2, FN3, FNBC, FN4). În cele ce urmează sunt date definiţiile formelor normale având în vedere dependenţele funcţionale.

O relaţie R este în prima formă normală (FN1) dacă şi numai dacă toate atributele sale iau numai valori atomice (nu pot fi descompuse). Spre exemplu, atributul Adresa ar putea fi considerat un atribut neatomic dacă în cadrul adresei ne-ar interesa localitatea, strada etc., caz în care trebuie descompus în atribute atomice.

O relaţie R este în a doua formă normală (FN2) dacă este în FN1 şi orice atribut neprim este total dependent faţă de orice cheie a relaţiei (atributele prime sunt atribute care fac parte dintr-o cheie a relaţiei şi cele neprime sunt atributele care nu aparţin nici unei chei a relaţiei).

O relaţie R este în a treia formă normală (FN3) dacă este în FN2 şi nici un atribut neprim nu este funcţional dependent faţă de un alt atribut neprim al relaţiei.

O relaţie R se află în forma normală Boyce-Codd (FNBC) dacă singurele dependenţe funcţionale admise sunt cele în care o cheie determină un alt atribut (nici un atribut prim sau neprim nu poate fi dependent funcţional faţă de un alt atribut dacă acesta nu este sau nu conţine o cheie).

Dependenţe multivalorice

Pentru ilustrarea acestui tip de dependenţe se ia în considerare următoarea schemă de relaţie:

Clase(Clasa, Discipline, Elevi)

ce conţine clasele dintr-o instituţie de învăţământ, iar pentru fiecare clasă sunt înregistrate disciplinele ce se predau şi elevii înmatriculaţi în clasa respectivă. Se poate constata că relaţia *Clase* poate rezulta prin operaţia de cuplare după atributul *Clasa* a următoarelor două relaţii:

CD(Clasa, Discipline)

CE(Clasa, Elevi)

În relaţia *Clase*, presupunând că pentru o clasă dată, fiecare elev frecventează toate disciplinele înregistrate pentru acea clasă, există dependenţele multivalorice:

Clasa ->> Discipline

Clasa ->> Elevi.

Ca şi în cazul dependenţelor funcţionale, existenţa dependenţelor multivalorice prezintă aceleaşi neajunsuri privind redundanţa datelor şi anomalii la efectuarea operaţiilor de adăugare, actualizare şi ştergere înregistrări în baza de date.

O relaţie R este în a patra formă normală dacă singurele dependenţe multivalorice admise sunt cele determinate de un alt atribut care este o cheie sau care conţine o cheie a relaţiei.

Întrucât orice dependenţă funcţională este un caz particular de dependenţă multivalorică, rezultă că orice relaţie care se află în forma normală FN4, se află şi în forma normală FNBC. Transformarea unei relaţii într-o colecţie de relaţii care să se afle în FN4 este similară cu trecerea în FNBC, însă trebuie avută în vedere atât eliminarea dependenţelor funcţionale cât şi a dependenţelor multivalorice.

În concluzie, putem afirma că în cazul formelor normale de la FN1 la FN4, trecerea de la o formă normală la alta s-a făcut prin descompunerea unei relaţii în altele două, urmărindu-se eliminarea dependenţelor funcţionale şi multivalorice. O relaţie aflată în forma normală FN4 nu mai poate fi descompusă în continuare pe baza acestei metode. Există situaţii când relaţii aflate în FN4 conţin redundanţe şi prezintă anomalii la operaţiile de adăugare, ştergere şi actualizare. Aceste anomalii sunt cauzate de existenţa dependenţelor de cuplare şi pot fi eliminate prin descompunerea relaţiei în 3 sau mai multe relaţii a căror cuplare are ca rezultat relaţia iniţială.

Dependenţe de cuplare

Se consideră schema de relaţie:

SDS (Specializari, Discipline, Studenti)

care conţine disciplinele care se predau la diverse specializări şi studenţii care le frecventează, cu precizarea că pot exista discipline opţionale care nu sunt frecventate de toţi studenţii de la specializarea respectivă. În aceste condiţii în cadrul schemei de relaţie SDS nu au loc dependenţele multivalorice:

Specializari ->> Discipline

Specializari->> Studenti

ceea ce înseamnă că relaţia SDS este în FN4. Deşi este în FN4, relaţia SDS conţine mai multe redundanţe care pot conduce la anomalii de actualizare. Pe de altă parte, relaţia SDS nu poate fi descompusă în două componente din a căror cuplare să rezulte relaţia iniţială cu conservarea informaţiei. Se constată însă că relaţia SDS poate fi descompusă în următoarele 3 relaţii:

SD(Specializari, Discipline)

SS(Specializari, Studenti)

DS(Discipline, Studenti)

şi relaţia SDS este rezultatul cuplării relaţiilor: SD, SS şi DS fără pierdere de informaţie.

SDS = SD►◄SS►◄DS.

În acest caz spunem că în relaţia SDS există o dependenţă de cuplare. Dependenţele multivalorice sunt cazuri particulare de dependenţe de cuplare.

A cincea formă normală este o generalizare a formei normale patru, trecerea unei relaţii în FN5 presupunând eliminarea dependenţelor de cuplare existente în cadrul relaţiei, împreună cu anomaliile pe care acestea le creează. În cadrul unei relaţii pot exista dependenţe de cuplare care nu conduc la redundanţă în memorarea datelor şi nu produc anomalii la operaţiile efectuate asupra înregistrărilor bazei de date (acestea sunt dependenţele de cuplare implicate de o cheie a relaţiei).

Dependenţa de cuplare este o proprietate ce garantează că nu se generează înregistrări false la reunirea relaţiilor obţinute prin descompunere. O relaţie se află în forma normală cinci dacă ea nu poate fi descompusă în alte relaţii fără a pierde informaţie.

O relaţie este în forma normală cinci (FN5) dacă şi numai dacă toate dependenţele de cuplare existente în relaţie sunt implicate de o cheie a acesteia. Relaţia SDS se poate descompune, cu conservarea conţinutului de informaţie, în cele 3 componente ale sale: SD, SS şi DS care sunt în FN5.

Având în vedere similaritatea ce există între definiţiile pentru FNBC, FN4 şi FN5, acestea pot fi unificate în următoarea definiţie [20]:

O relaţie R este în FNBC**,** FN4, FN5 dacă şi numai dacă singurele dependenţe funcţionale, multivalorice, de cuplare existente sunt cele implicate de o cheie a relaţiei R.

În concluzie, prin procesul de normalizare se realizează eliminarea din schemele de relaţie a dependenţelor (funcţionale, multivalorice şi de cuplare) cu scopul de a obţine o schemă relaţională mai bună din punctul de vedere al redundanţei datelor şi al anomaliilor ce pot apare la operaţiile de adăugare, ştergere şi actualizare înregistrări în baza de date. Normalizarea unei scheme de relaţie R înseamnă înlocuirea acesteia cu o mulţime de proiecţii R1,...,Rn astfel încât R să fie echivalentă cu uniunea proiecţiilor R1,...,Rn. Deşi normalizarea este o operaţie utilă în proiectarea bazelor de date, aceasta nu oferă întotdeauna reţete pentru obţinerea celor mai bune modele şi de aceea este la latitudinea proiectantului decizia de a aplica sau nu o anumită etapă de normalizare după o analiză temeinică a avantajelor şi dezavantajelor modelului obţinut. În unele cazuri normalizarea completă, până la FN5, s-ar putea să fie dezavantajoasă. Având în vedere constatările de mai sus se poate afirma că deşi normalizarea nu reprezintă o soluţie general valabilă în orice situaţie, totuşi dacă pentru proiectarea bazei de date se aplică corect o metodologie de proiectare descendentă, modelul rezultat va fi de la sine normalizat. Cercetările în acest domeniu continuă, fiind definite şi alte forme normale printre care FN6 pentru baze de date temporale. O bază de date temporală, pe lângă datele curente, conţine şi date istorice, iar factorul (atributul) timp are un rol esenţial (exemple concludente de astfel de baze de date sunt depozitele de date). Astfel, în proiectarea unei baze de date temporale trebuie avute în vedere şi alte operaţii de descompunere a schemelor de relaţie şi anume:

- descompunerea orizontală – pentru separarea datelor curente de datele istorice;
- descompunerea verticală – pentru separarea atributelor aceleiaşi entităţi având în vedere valorile lor raportate la atributul temporal.

În proiectarea unei baze de date nu este exclusă nici operaţia inversă normalizării numită denormalizare [16], prin care se urmăreşte înlocuirea unei colecţii de scheme de relaţie cu o schemă de relaţie echivalentă în vederea eliminării necesităţii efectuării unor operaţii de cuplare care pot fi costisitoare. Dacă în cazul normalizării tendinţa este de a ajunge la nivele cât mai înalte (FN5), pentru denormalizare nu există criterii clare putând fi avute în vedere doar aspecte legate de performanţele anumitor aplicaţii.

Un alt principiu care se urmăreşte în proiectarea unei baze de date este principiul proiectării ortogonale conform căruia în cadrul unei baze de date două scheme de relaţie reale (variabile de relaţie de bază) nu trebuie să aibă semnificaţii suprapuse. În timp ce prin normalizare se urmăreşte reducerea redundanţei din cadrul unei scheme de relaţie, prin proiectarea ortogonală se urmăreşte reducerea redundanţei dintre schemele de relaţie.