|  |
| --- |
| Facultatea de automatică, calculatoare și electronică |
| EerieEdit |
| Ingineria Programării |
|  |
| **Alexandru Salan, Daniel Tilă, Laurențiu Nicola** |
| **CR 10304b** |

|  |
| --- |
|  |

# ANALIZA CERINŢELOR

## Descriere

Se cere crearea unei aplicatii in care se pot reprezenta diagrame Entitate –Relatie. Acest model a fost introdus de Peter Chen în 1976 si se bazeaza pe trei concepte fundamentale:

1. **Entitate** – orice obiect, fenomen sau concept care poate fi deosebit de alte obiecte, fenomene sau concepte asemănătoare prin valorile unor caracteristici numite atribute; Se numeşte entitate tip (clasă de entităţi) mulţimea entităţilor cu aceaşi structură. Entitatile se caracterizeaza prin nume si semantica
2. **Atributul** – este orice caracteristică a uneientităţi, si caracterizeaza prin nume, structura si valoare.
3. **Legatura** – este o corespondenţă ce poate fi pusă în evidenţă între entităţile aceleaşi entităţi tip ori entităţile a două sau mai multe entităţi tip pe baza unui criteriu bine definit.

Aplicatia se doreste a fi folosita inclusiv in scop didactic de catre studentii facultatii de Automatica Calculatoare si Electronica, pentru intelegerea si crearea de de astfel de diagrame. Principalele noastre obiective au fost sa realizam o aplicatie simpla de folosit si cu o interfata ergonomica.

## Domeniu

Acest proiect se adreseaza tuturor persoanelor care doresc sa foloseasca modelul entitate-relatie pentru proiectarea bazelor de date, in scop didactic, comercial sau personal.

## Actionari / Interese

Utilizatorii aplicatiei vor fi: profesor sau studenti, ambii avand ca interes utilizarea unei interfete grafice usor de folosit si rapida, cat si corectitudinea schemei realizate.

Aplicaţia utilizator este un subsistem software care mediază prin intermediul unei interfeţe grafice accesul utilizatorilor la diagramele entitate-relatie.

## Actori & Obiective

*Untilizatorul* are ca obiectiv folosirea cat mai rapida si usoaraa aplicatiei, fara a intampina probleme in exprimarea ideilor si fara a fie nevoit sa-si modifice scheme deja existente, putand sa le transpuna (atata timp cat respecta acelasi standard).

## Scenarii de utilizare

Urmatoarele cazuri for descrie modurile in care un utilzator poate folosi aplicatia pentru a realiza o schema.

### Se doreste crearea unei noi scheme

**a) Obiectiv/ Context**

Un student primeste ca tema pentru un laborator sa proiecteze o baza de date pentru un magazin folosind modelul entitate-relatie.

**b) Scenariu / Pasi**

1. Intelegerea cerintelor problemei

2. Gasirea unei implementari corecte

3. Deschiderea aplicatiei si crearea unui nou proiect

4. Implementarea schemei folosing interfata grafica

5.Aplicatie confirma ca datele au fost introduse corect si ca s-a respectat standardul.

6. Utilizatorul va salva proiectul facut pana la momentul respectiv intr-un fisier binar de pe disc, pentru a putea fi utilizat sau citit cu alta ocazie.

**c) Extensii**

Daca cumva datele nu sunt introduse corect sau nu sunt introduse aplicatia nu va permite crearea unui asfel de obiect. De asemenea aplicatia nu va lasa ca un atribut sa ramana nelegat de o entitate sau o relatie.

**d) Tehnologii / Cerinte**

Aplicatia nu va permite utilizatorului sa nu respecte standardul prezentat anterior.

Datele sunt memorate intr-un fisier binar creat de catre aplicatie.

### Se doreste continuarea unei lucrari incepute anterior

1. **Obiectiv/Continut**

Un utilizator doreste sa continue un proiect neterminat.

1. **Scenariu/ Pasi**
2. Deschiderea aplicatiei
3. Deschiderea fisierului in care a salvat proiectul anterior
4. Continuarea impementarii schemei folosind interfata grafica
5. Validarea datelor introduse de catre aplicatie .
6. Salvarea proiectului pentru reutilizare
7. **Extensii**

Aplicatia va salva in acelasi fisier pe care l-a deschis, supra scriind informatia. Pentru a evita suprascrierea se poate alege sa se salveze intr-un alt fisier

# Proiectare. Arhitectura

## Structura subsistemelor

Programul este impartit in doua parti:

1. **Partea de interactiune cu utilizatorul**, aceasta cuprinde desenarea obiectelor si pozitionarea lor cat si controalele pentru mouse si pentru tastatura care sunt gandite sa fie cat mai usor de utilizat
2. **Parta logicii de aplicatie**, care se ocupa cu corectitudinea aplicatiei, ea va verifica daca nu se incalca o regula, si se va ocupa de salvarea (incarcarea) din fisier.

## Modelarea controlului

Modelul controlului ales pentru realizarea proiectului este cel al al ferestrelor de dialog, ele sunt folosite pentru a crea si modifica obiecterele entitatile, relatiile si atributele cat si pentru a confirma daca se doreste stergerea.

Interfata grafica interactioneaza cu utilizatorul prin intermediul ferestrelor de dialog si a toolbarului, programul memoreaza datele in RAM, iar cand i se cere sa se scrie intr-un datele el va scrie structura claselor din memorie in fisier.

## Descompunerea subsistemelor

Proiectul se poate descompune in doua subsisteme, interfata grafica si aplicatia in sine.

## Identificarea principalelor obiecte ale sistemului

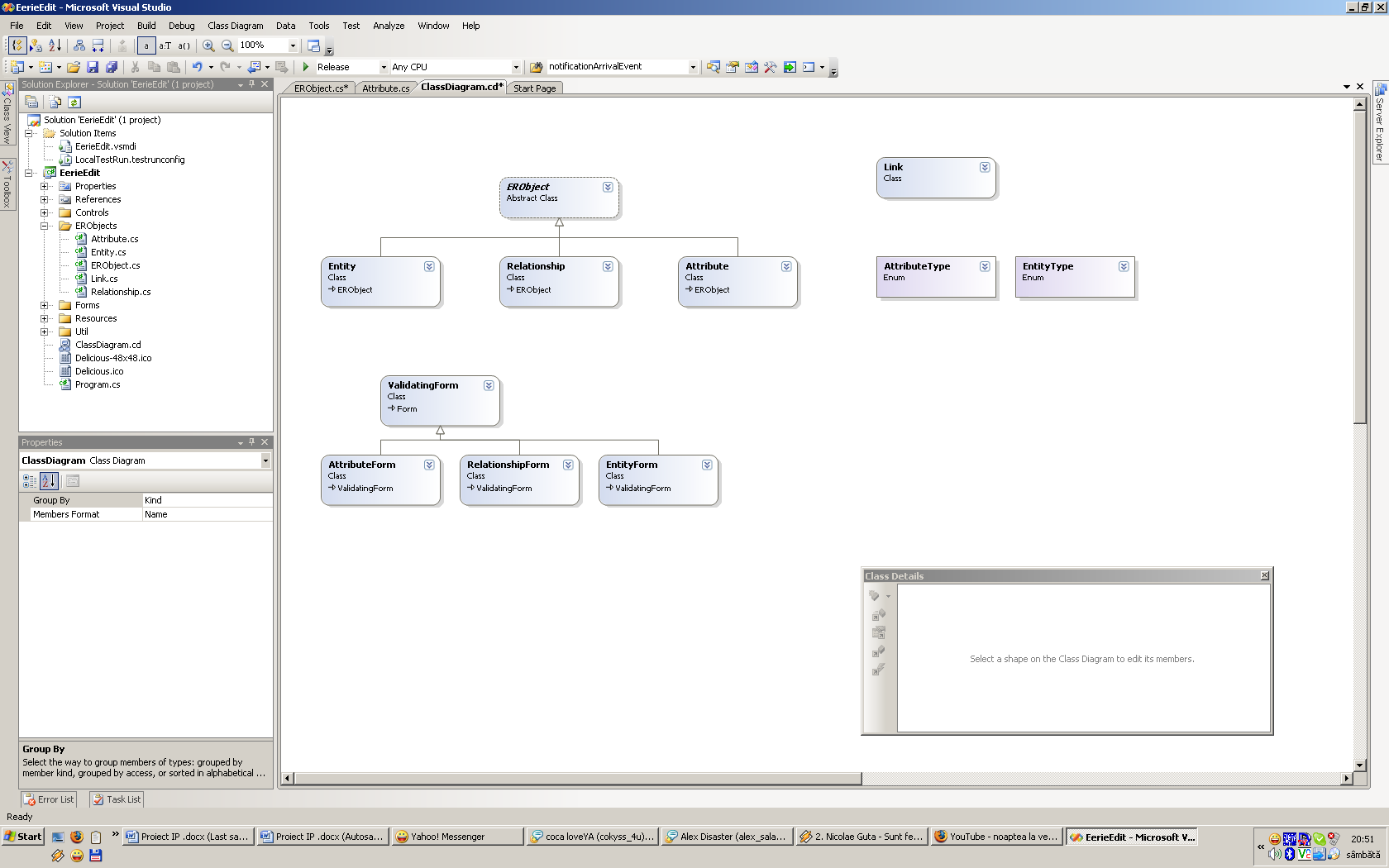
1. Obiectele legate de interfaţa grafică: *MainForm;*
2. Obiectele legate de aplicaţia utilizator: *MainForm*, DiagramEditor*, TablesViewer, AttributeForm, RelationshipForm, EntityForm;*

## Interfata grafica si programarea orientata pe obiecte

Acestea cuprind forma principala, si cele pentru adaugare de obiecte. Pentru fiecare obiect entitate-relatie se gaseste un buton specific in toolbar. La apasarea acestuia, apare o fereastra cu optiunile pentru obiectul care doriti sa-l adaugati. Adaugarea unui obiect propriu-zis se confirma cu butonul OK. Dupa apasarea acestuia va aparea in controlul de desenat marginea obiectului in timpul deplasarii mouselui.

Desenarea obiectelor este realizata asupra unui control care este pe toata suprafata casetei de dialog. Acesta se ocupa cu captarea evenimetelor de mouse, adaugarea obiectelor, stergerea acestora, creearea de legaturi, salvarea sau incarcarea diagramei, etc.

Controlul de desenare realizeaza desenarea a unor atribute, entitati si legaturi precum si a unor relatii. Entitatile, atributele si relatiile sunt clase separate care cuprind proprietati specifice si deriveaza din o clasa de baza ERObject. Aceasta cuprinde metode comune tuturor obiectelor precum enumerarea veciniolor, intersectiile cu obiecte, calcularea legaturilor etc. Ierarhia claselor arata precum in desenul urmator:



Realizarea programului folosind aceastra structura, ofera acestuia o flexibilitate in adaugarea de obiecte pe care acesta sa le deseneze, fiind de ajuns sa creezi o noua clasa care sa implementeze ERObject, si analiza situatiilor privind legatura cu celelalate obiecte. Mai important, poate, aceasta separare duce la modularizare si la separarea atributiilor.

Crearea entitatilor si a legaturilor dintre ele ar fi inutila fara tragerea unor concluzii privind modelul relational al unei baze de date, astfel ca programul are implementat un control aditional folosit pentru generarea tabelelor. Acesta primeste ca date de intrare legaturile creeate anterior si permite generarea tabelelor pentru baze de date.

# Concluzii

Prin cele prezentate s-a putut urmari o mica parte a eforturilor depuse pentru dezvoltarea aplicatiei in cauza. Desi este departe de a fi terminata, prioritizarea functionalitatii de baza a permis crearea unui produs utilizabil inca din fazele timpurii ale realizarii.