

Ingineria Programării

Cursul 4 – 9 Martie 2020

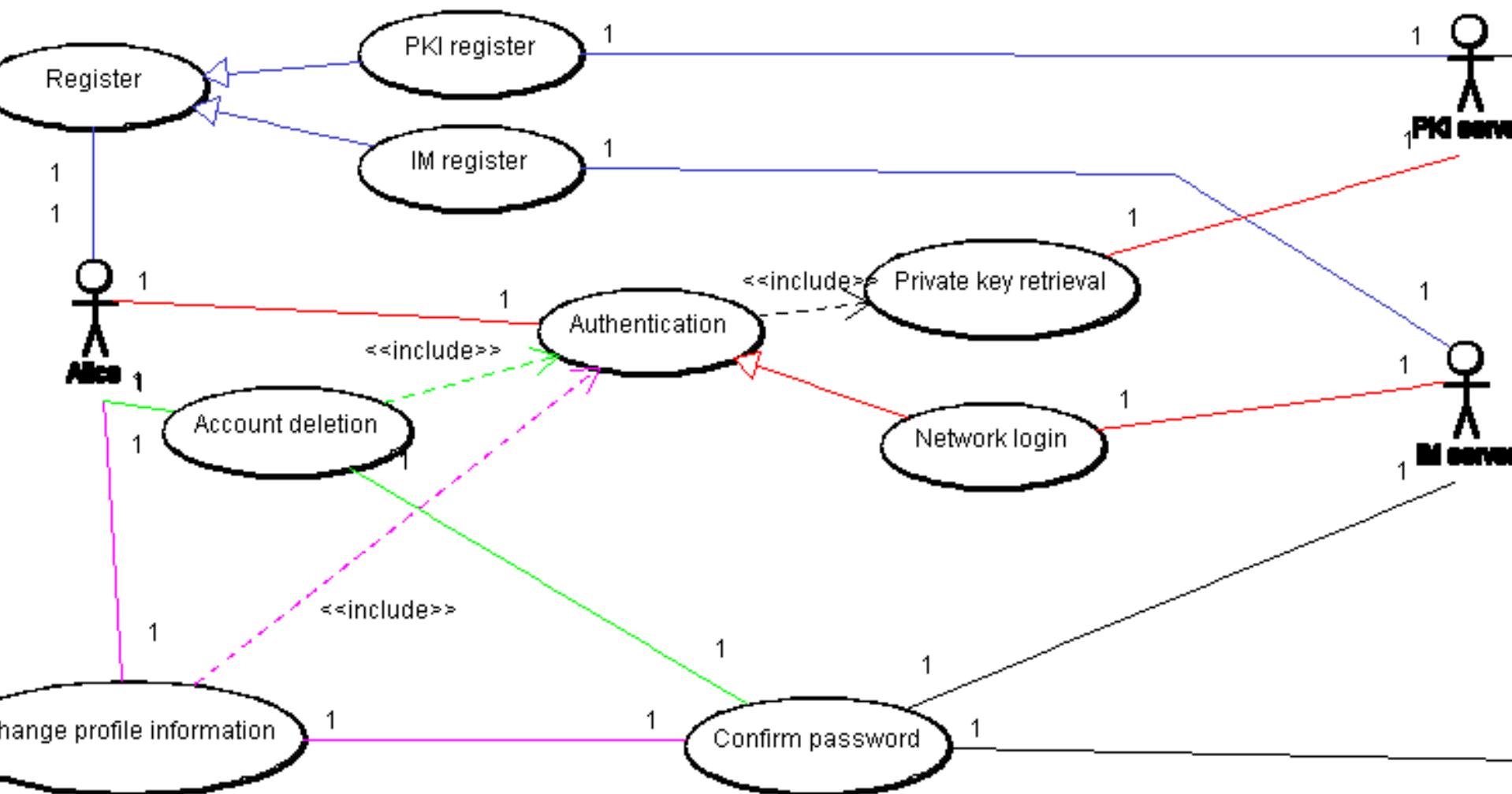
adiftene@info.uaic.ro

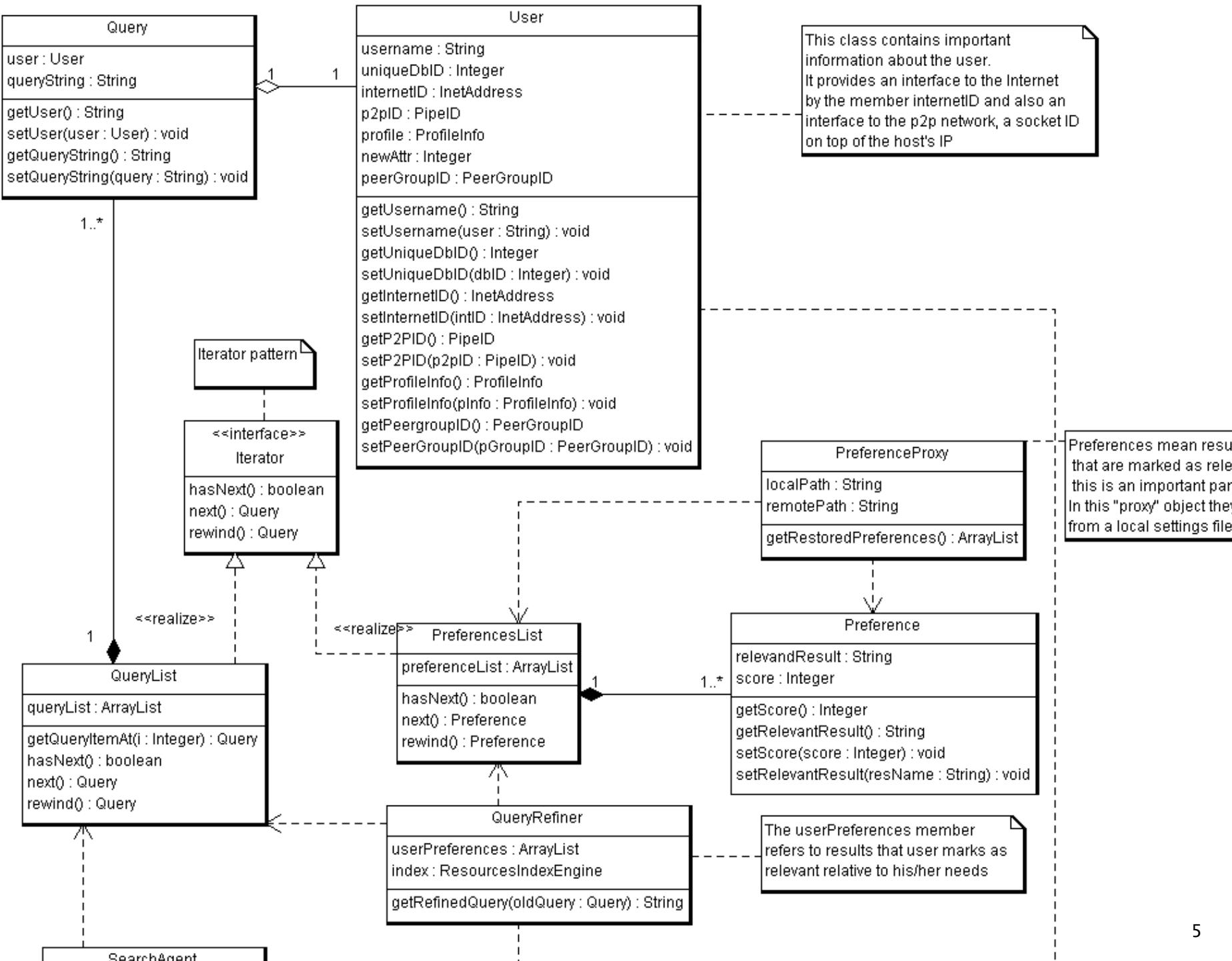
Cuprins

- ▶ Din Cursurile trecute...
- ▶ Diagrame UML – Exemple
- ▶ C4 Model
- ▶ Forward & Reverse Engineering

Din cursurile trecute...

- ▶ Diagrame
- ▶ Diagrame UML
- ▶ Diagrame Use Case
- ▶ Diagrame de Clase





UML2.0 – 13 Tipuri de Diagrame

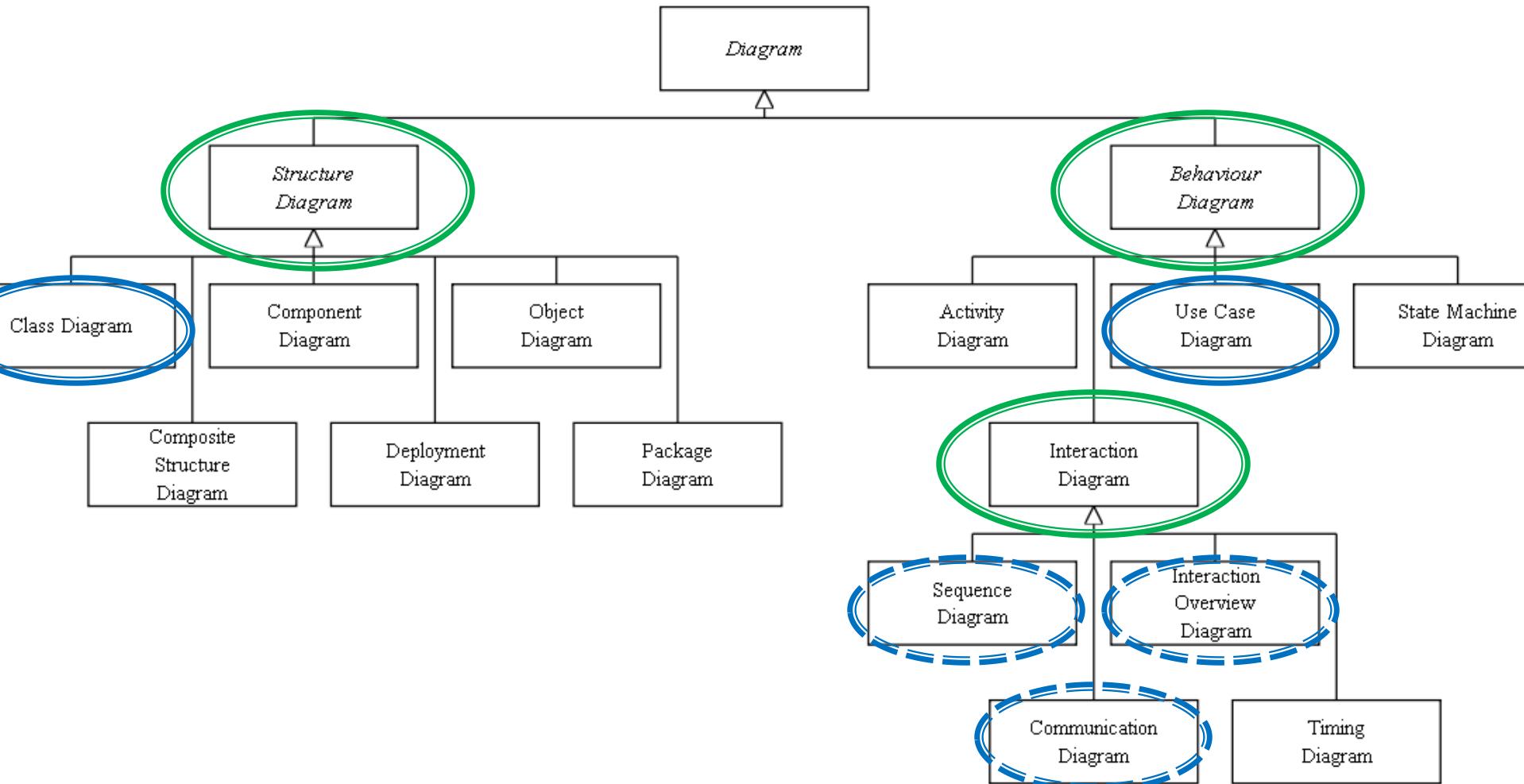
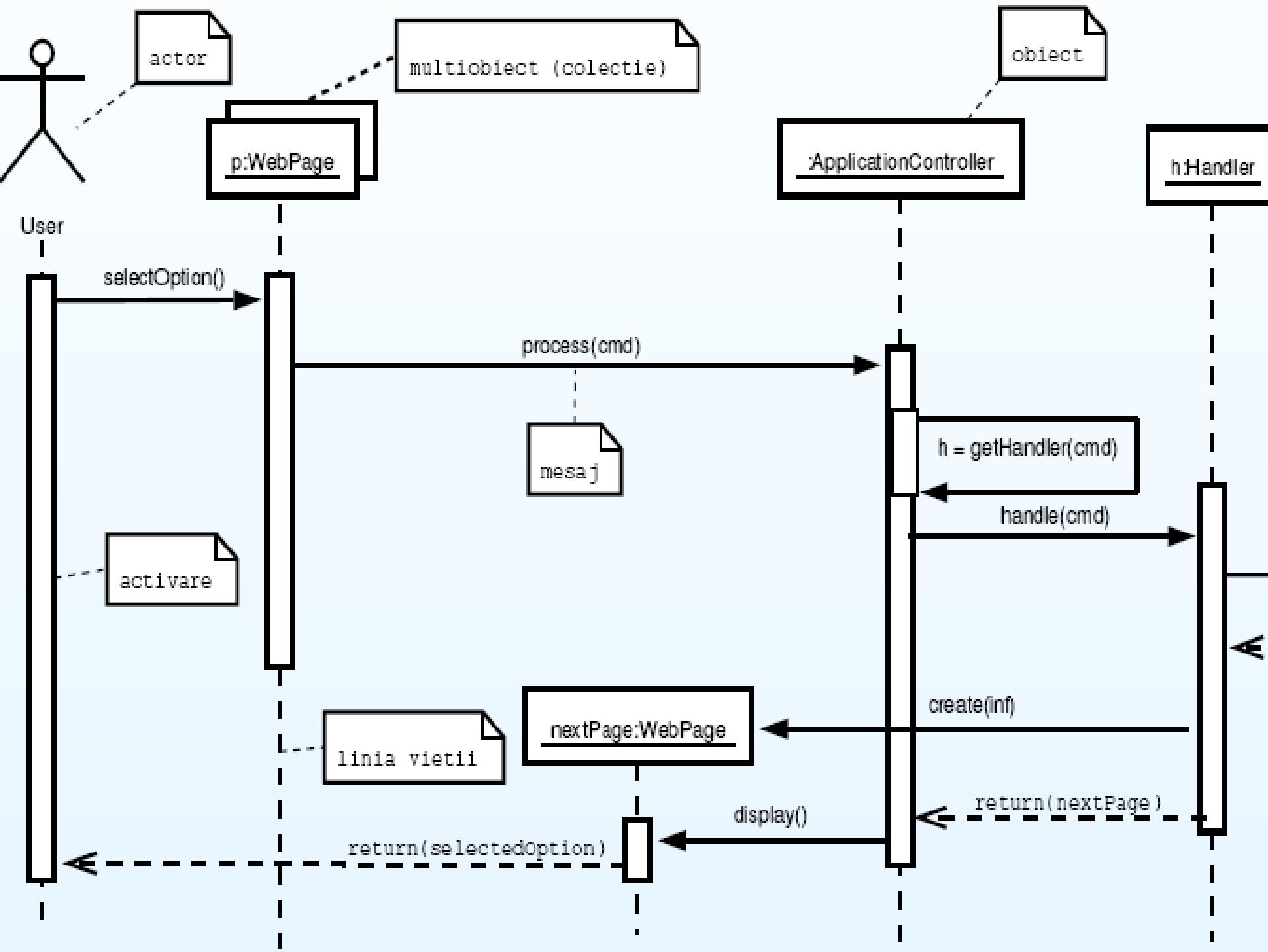
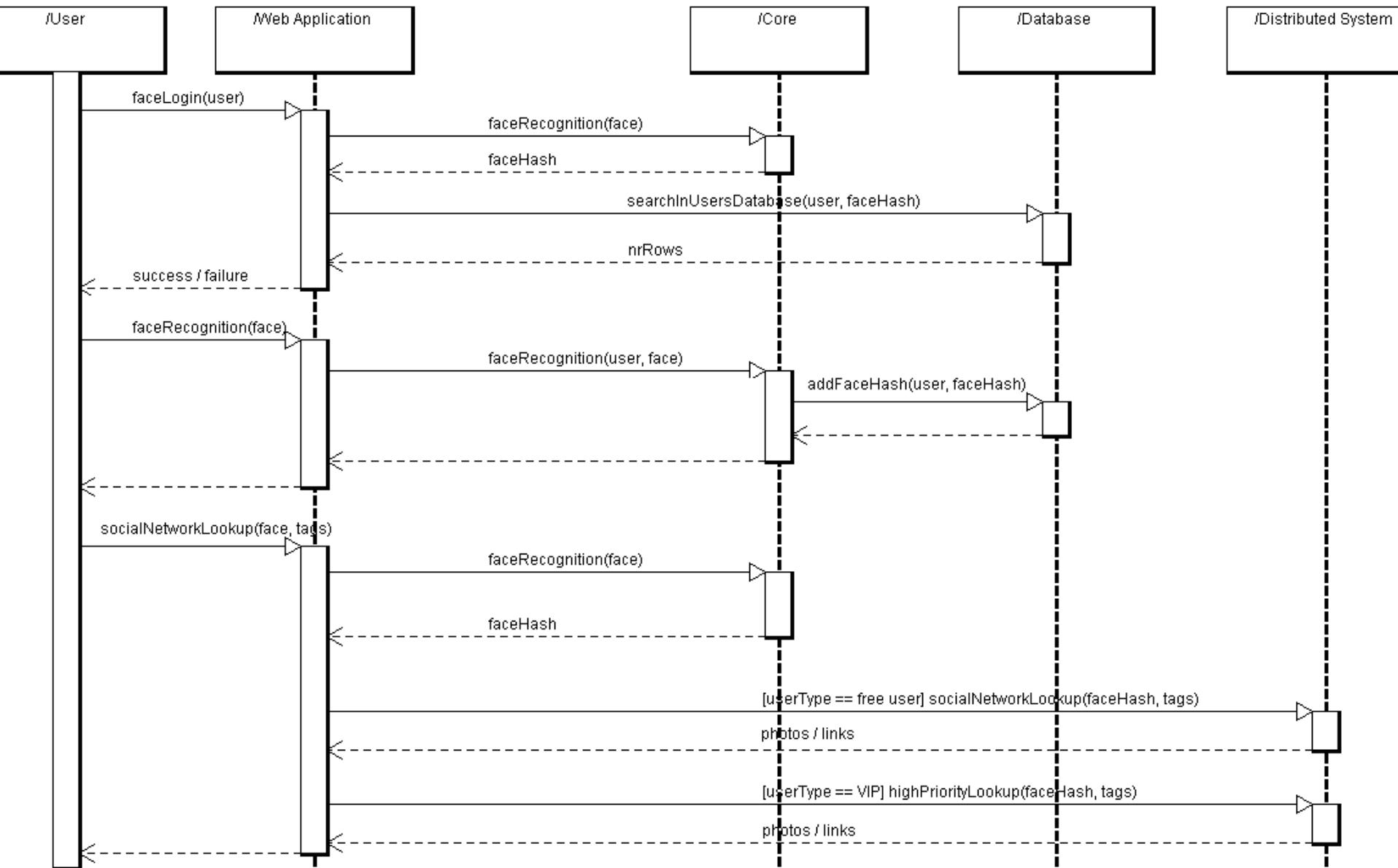


Diagrama de Secvență

- ▶ **Diagrama de secvență** cuprinde secvența acțiunilor care au loc în sistem, invocarea metodelor fiecărui obiect ca și ordinea în timp în care aceste invocări au loc
- ▶ O diagramă de secvență este bidimensională
 - Pe axa verticală se prezintă viața obiectului
 - linia vieții obiectelor (grafic: linie punctată)
 - perioada de activare în care un obiect preia controlul execuției (grafic: dreptunghi pe linia vieții)
 - Pe axa orizontală se arată secvența creării sau invocărilor
 - mesaje ordonate în timp (grafic: săgeți)

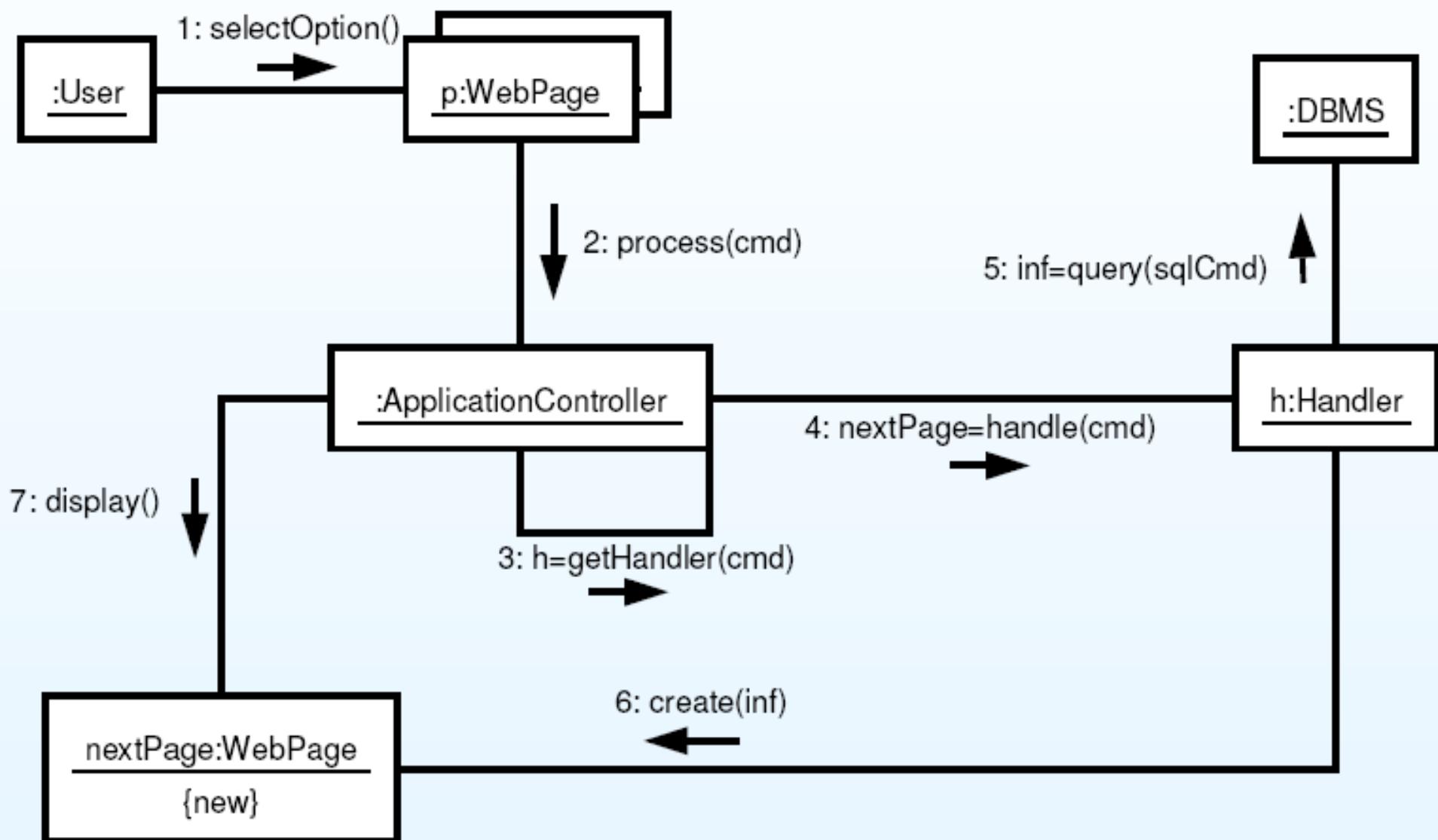




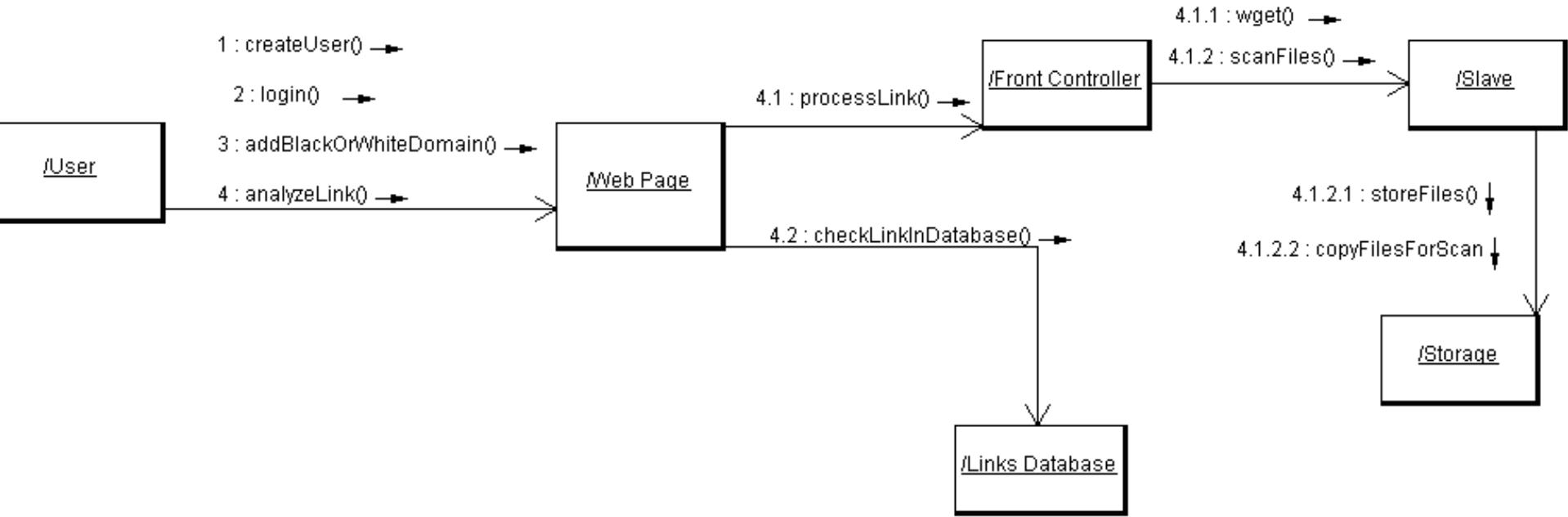
Diagramă de Colaborare

- ▶ Pune accentul pe organizarea structurală a obiectelor care participă la interacțiune
- ▶ Ilustrează mai bine ramificări complexe, iterări și comportament concurrent
- ▶ Poate conține:
 - Obiecte, clase, actori
 - Legături între acestea
 - Mesaje

Exemplul 1



Exemplul 3

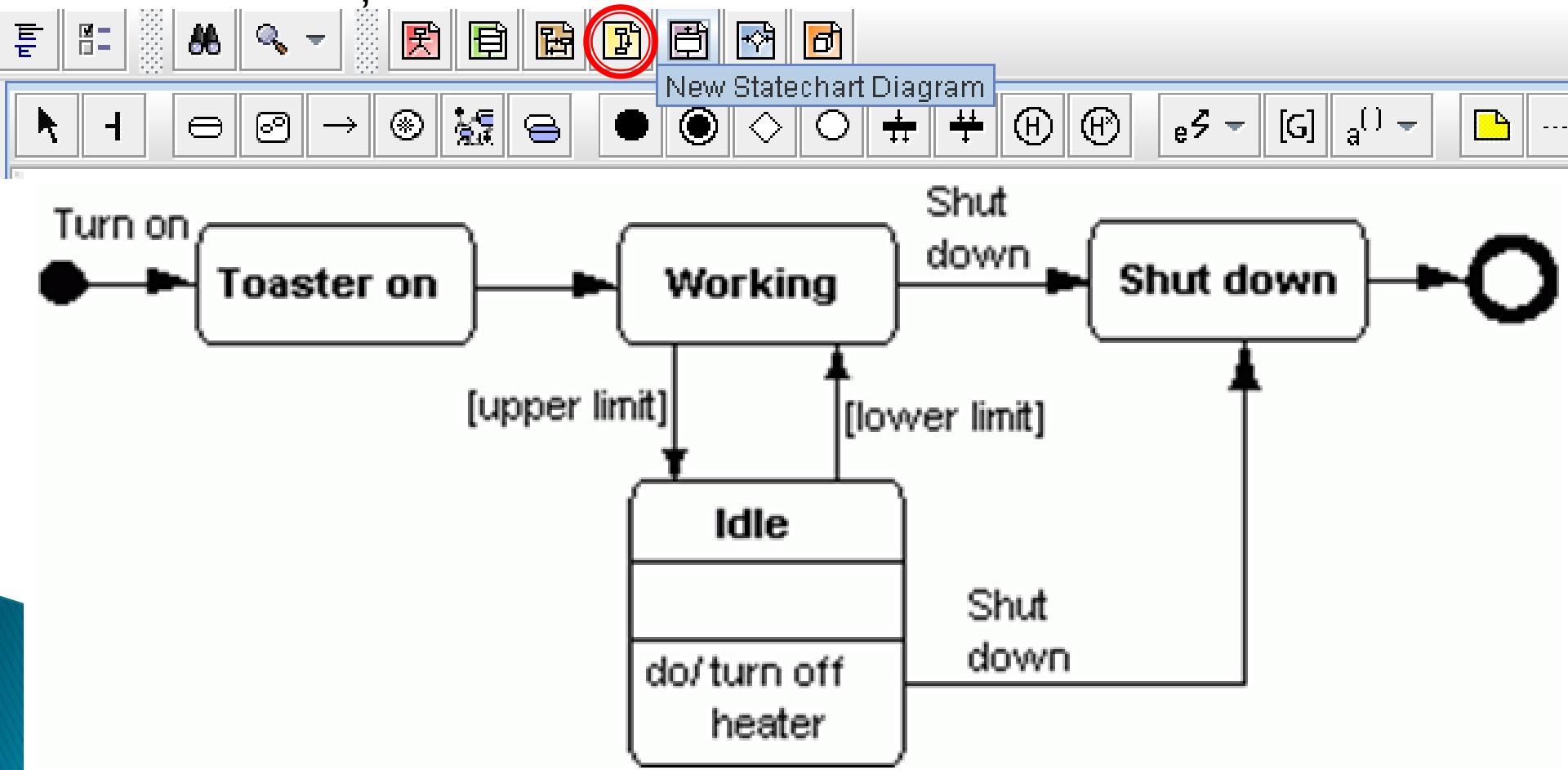


Diagrame comportamentale

- ▶ Diagrame de stări, diagrame de activități
- ▶ Elemente de bază
 - Eveniment
 - Acțiune
 - Activitate

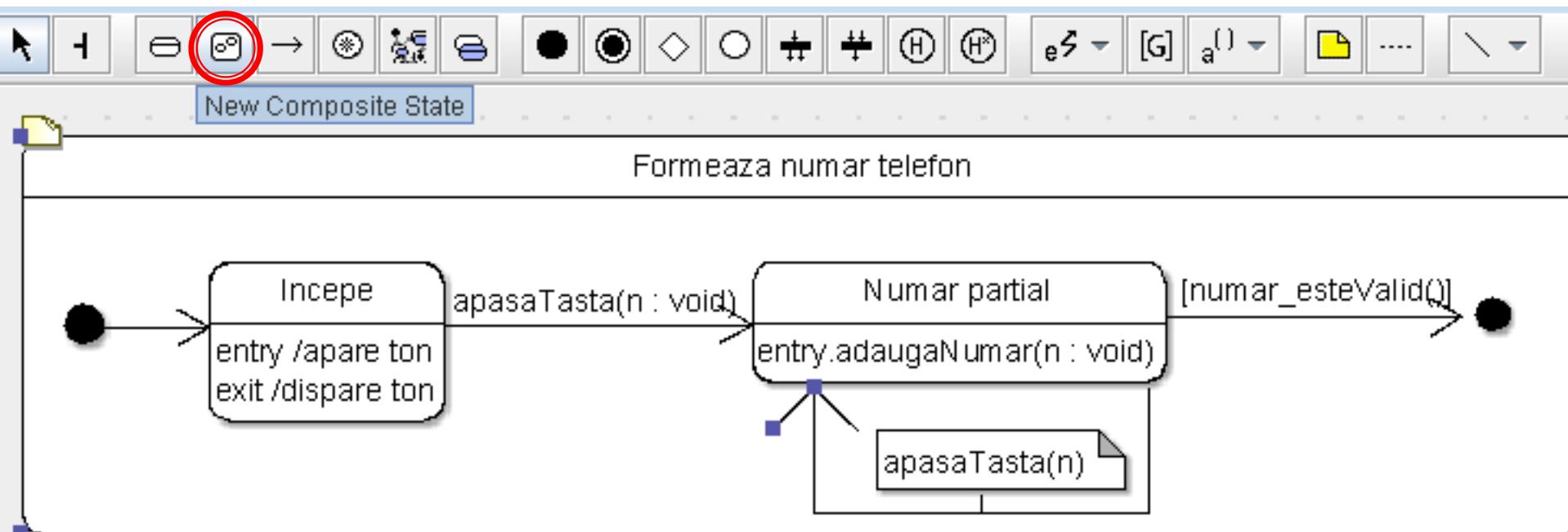
Diagramă de Stări

- ▶ Contine:
 - Stări
 - Tranzitii



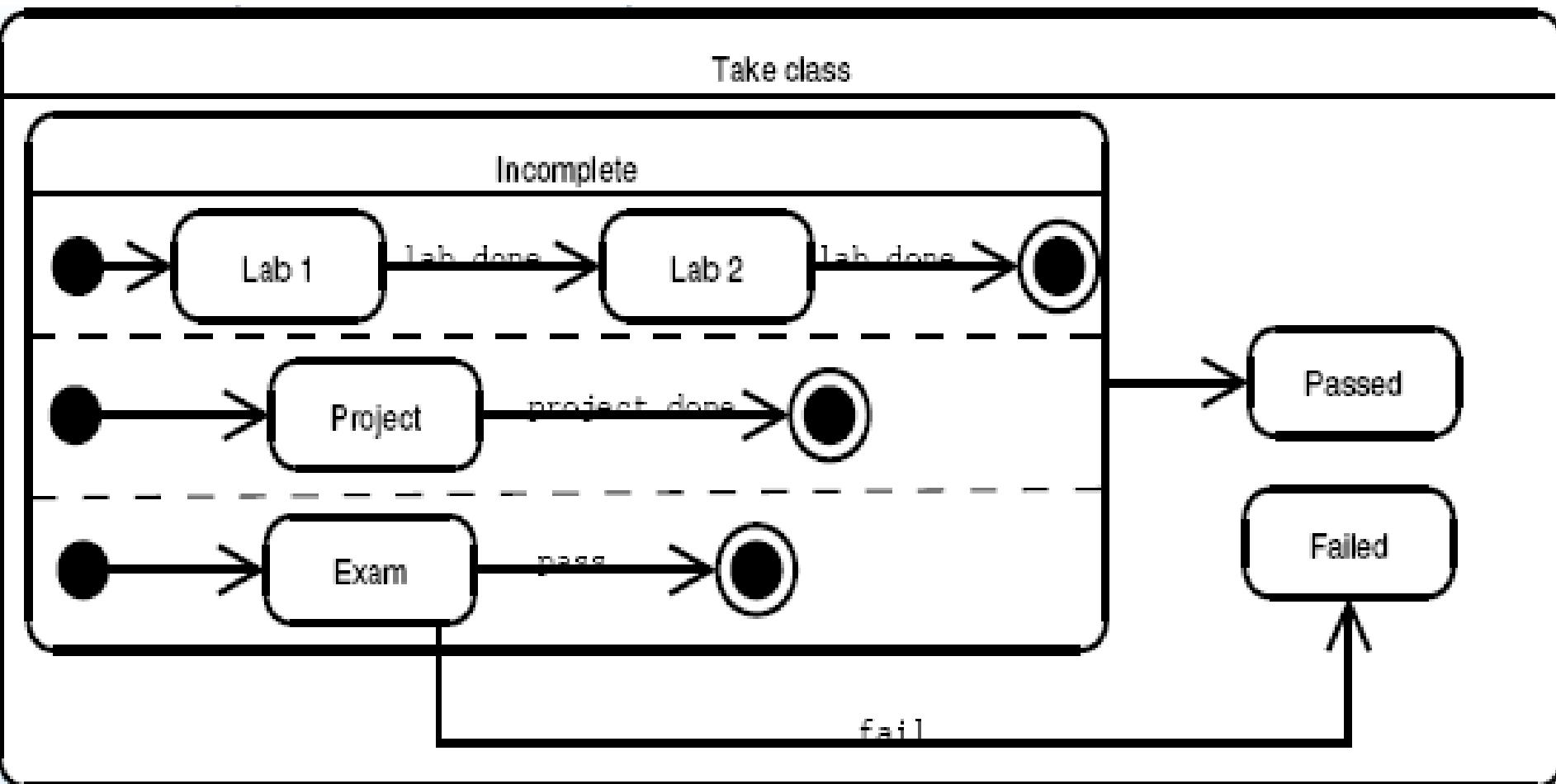
Exemplu de Stare compusă 1

- ▶ Stare compusă cu substări secvențial active:



Exemplu de Stare compusă 2

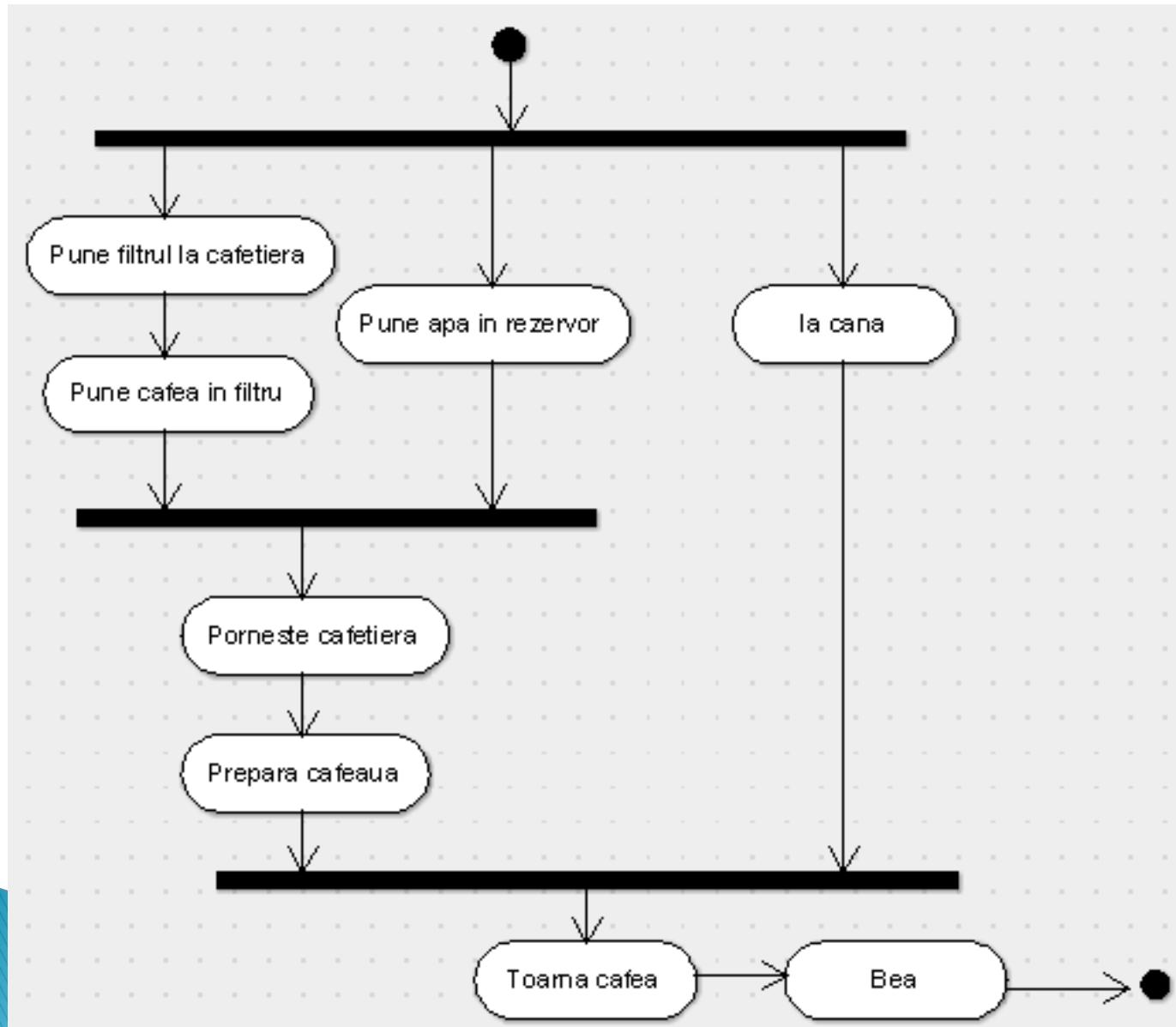
- ▶ Stare compusă cu substări paralel active:

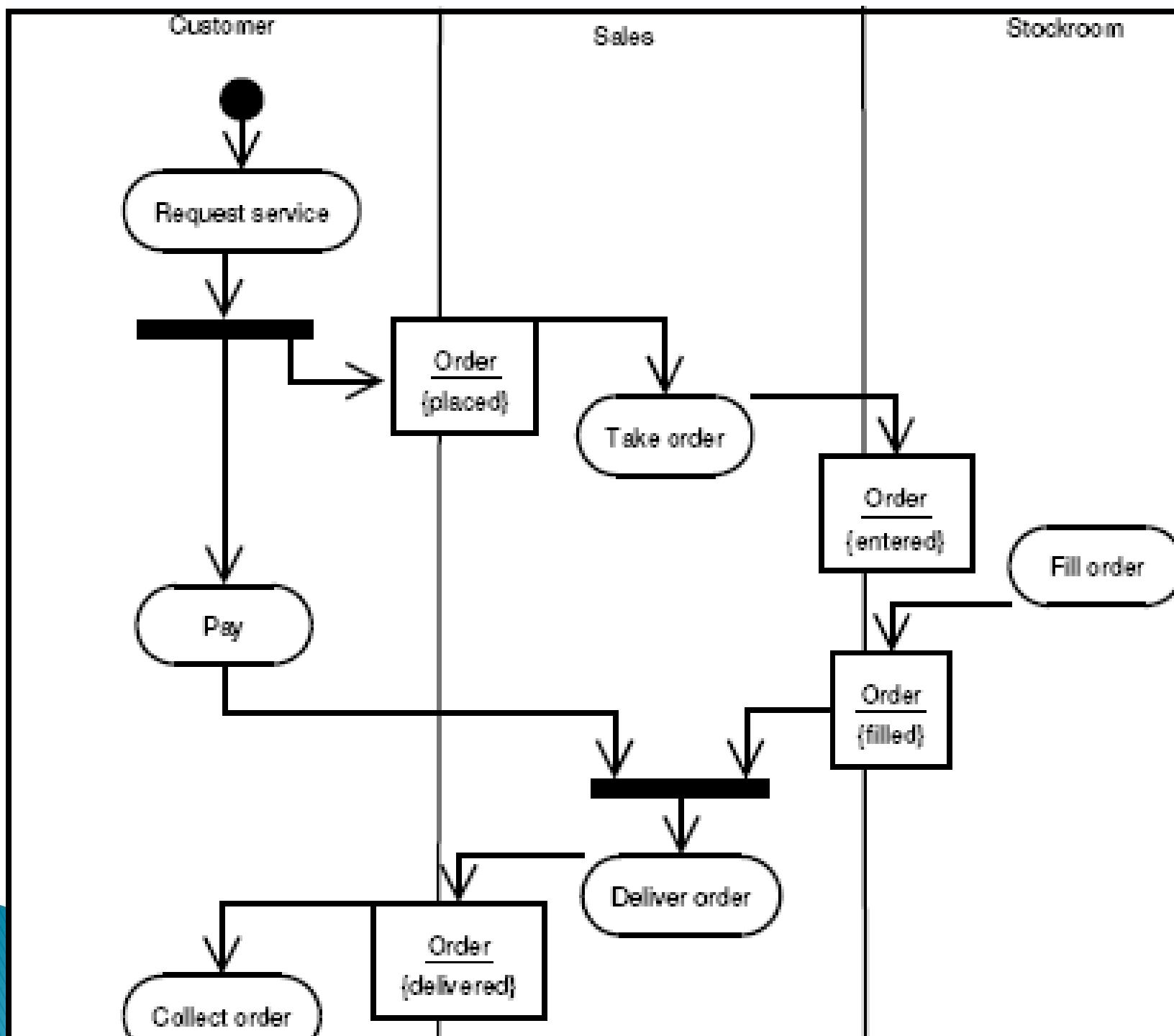


Diagramă de Activități (Activity Diagram)

- ▶ Folosită pentru a modela dinamica unui proces sau a unei operații
- ▶ Evidențiază controlul execuției de la o activitate la alta
- ▶ Se atașează:
 - Unei clase (modelează un caz de utilizare)
 - Unui pachet
 - Implementării unei operații

Exemplu de DA (Sincronizare)

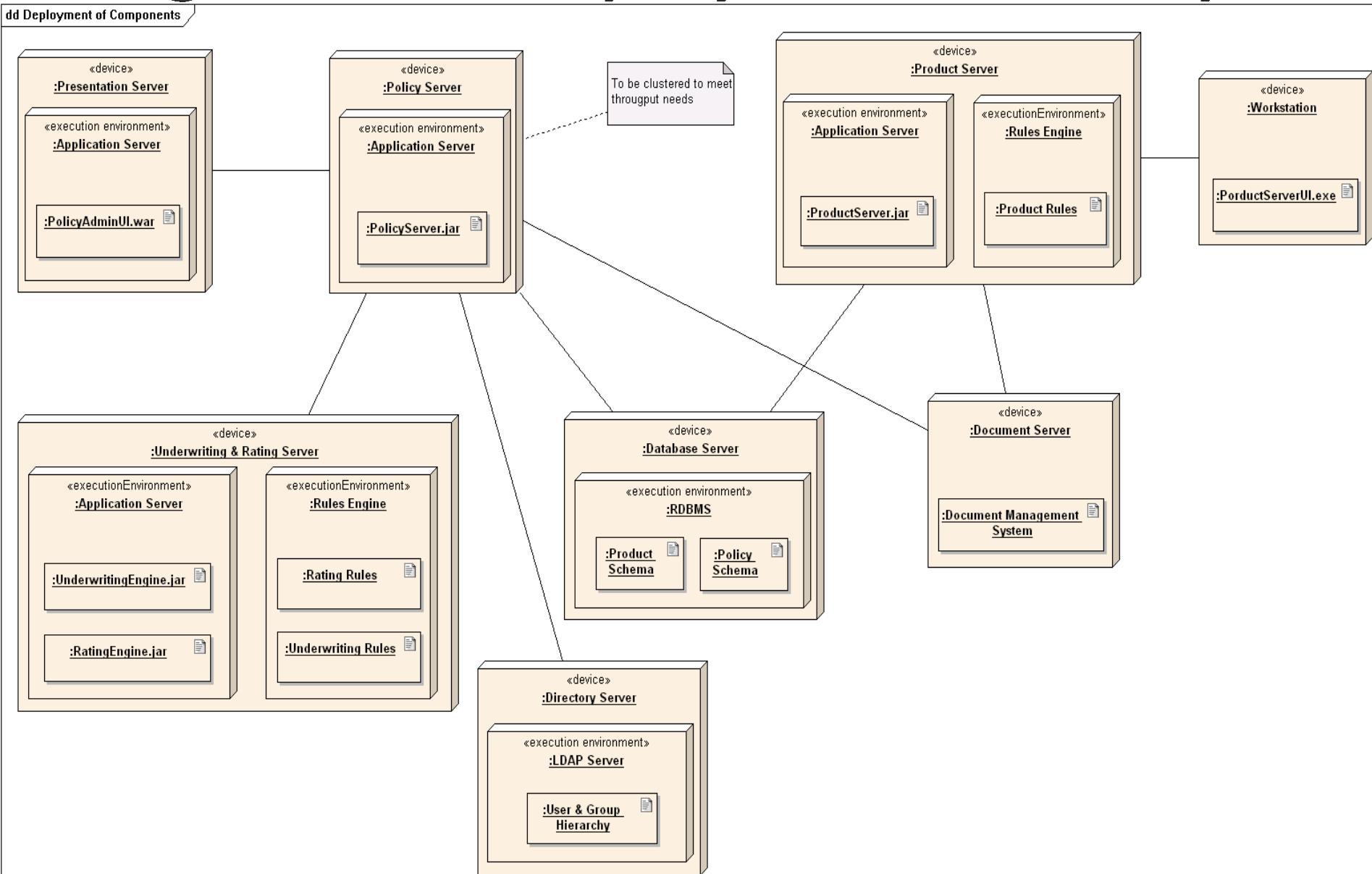




Diagrame de deployment

- ▶ Modelează mediul hardware în care va funcționa proiectul
- ▶ Exemplu: pentru a descrie un site web o diagramă de deployment va conține **componentele hardware**
 - server-ul web,
 - server-ul de aplicații,
 - server-ul de baze de date
- ▶ **Componentele software** de pe fiecare din acestea
 - Aplicația web
 - Baza de date
- ▶ Modul în care acestea sunt conectate:
 - JDBC, REST, RMI

Diagramă de deployment – Exemplu 1



Diagrame de Pachete (Package Diagram)

▶ Pachetul:

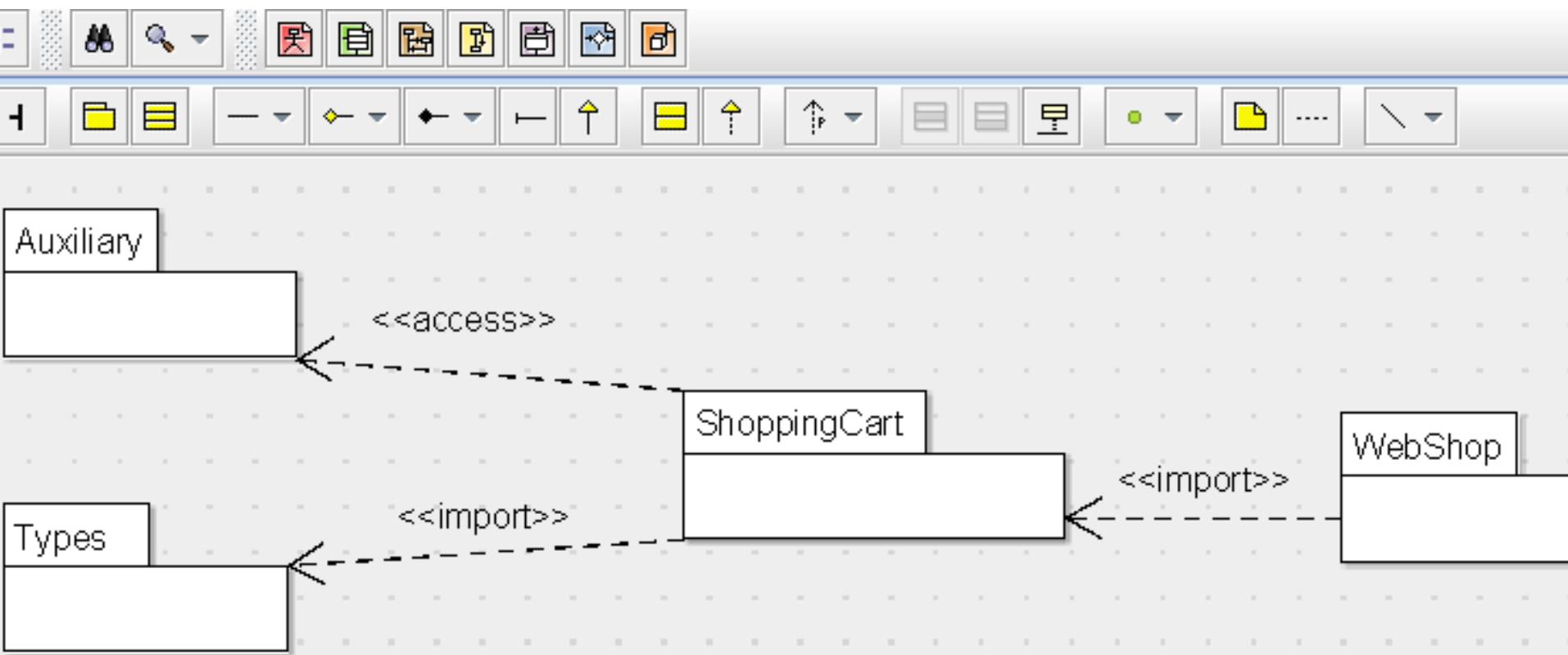
- Este un container logic pentru elemente între care se stabilesc legături
- Definește un spațiu de nume
- Toate elementele UML pot fi grupate în pachete (cel mai des pachetele sunt folosite pentru a grupa clase)
- Un pachet poate conține subpachete => se creează o structură arborescentă (similară cu organizarea fișierele/directoarelor)

Diagrame de Pachete 2

- ▶ Relații:
 - dependență <<access>> = import privat
 - dependență <<import>> = import public
- ▶ Ambele relații permit folosirea elementelor aflate în pachetul destinație de către elementele aflate în pachetul sursă fără a fi necesară calificarea numelor elementelor din pachetul destinație (similar directivei **import** din java)
- ▶ Aceste tipuri de diagrame se realizează în cadrul diagramelor de clasă

Exemplu de Diagramă de Pachete

- ▶ Elementele din Types sunt importate în ShoppingCart și apoi sunt importate mai departe de către WebShop
- ▶ Elementele din Auxiliary pot fi accesate însă doar din ShoppingCart și nu pot fi referite folosind nume necalificate din WebShop



Utilitatea diagramelor de pachete

- ▶ Împart sisteme mari în subsisteme mai mici și mai ușor de gestionat
- ▶ Permit dezvoltare paralelă iterativă
- ▶ Definirea unor interfețe clare între pachete promovează refolosirea codului (ex. pachet care oferă funcții grafice, pachet care oferă posibilitatea conectării la BD, etc...)

Recomandări în realizarea diagramelor UML

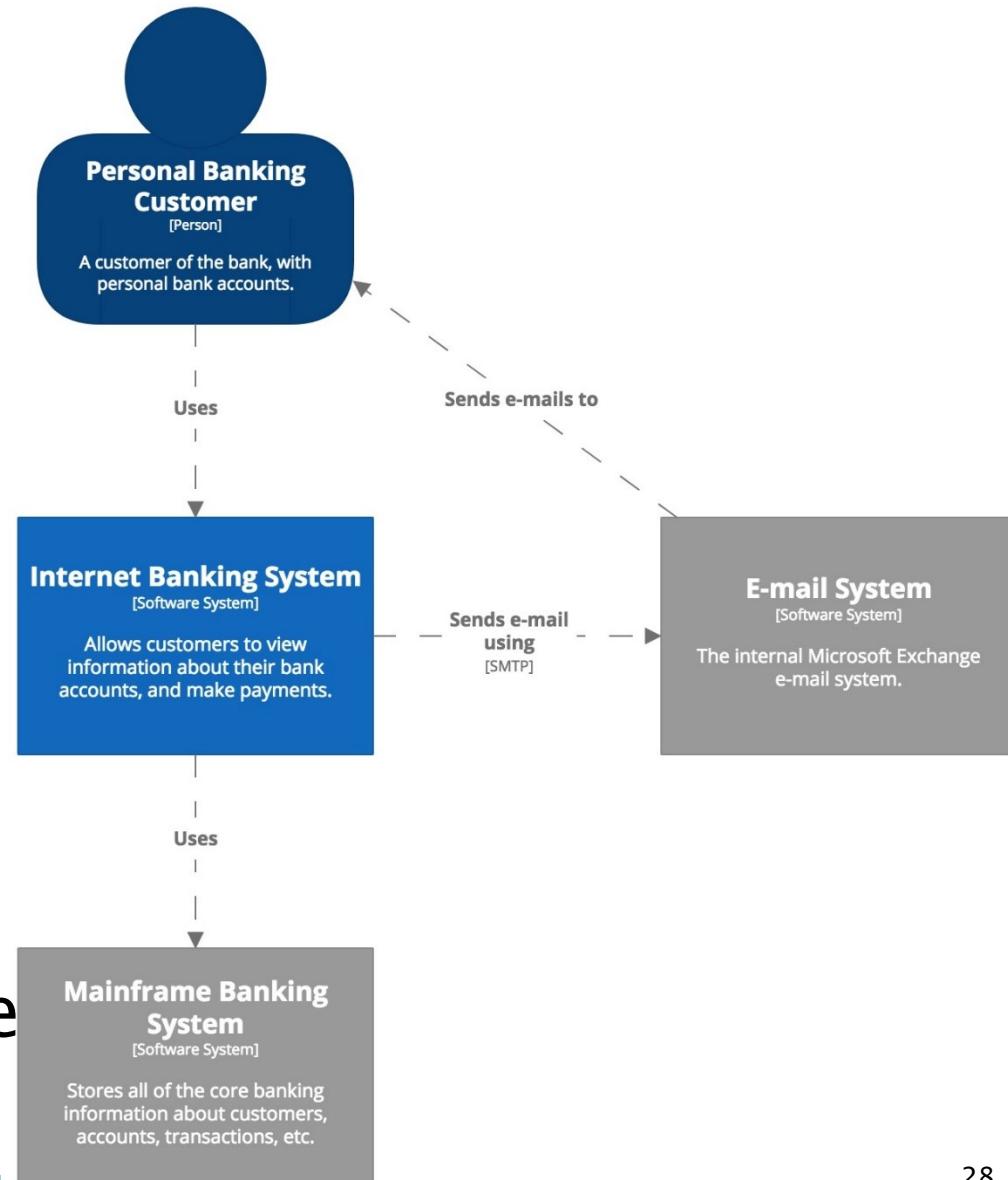
- ▶ Diagramele să nu fie nici prea complicate, dar nici prea simple: scopul este comunicarea eficientă
- ▶ Dați nume sugestive elementelor componente
- ▶ Aranjați elementele astfel încât liniile să nu se intersecteze
- ▶ Încercați să nu arătați prea multe tipuri de relații odată (evitați diagramele foarte complicate)
- ▶ Dacă este nevoie, realizați mai multe diagrame de același tip

The C4 Model for Software Architecture

- ▶ **Context, Containers, Components and Code**
- ▶ **Provides different levels of abstraction**, each of which is relevant to a different audience
- ▶ **Avoid ambiguity in your diagrams by including a sufficient amount of text as well as a key/legend for the notation you use**

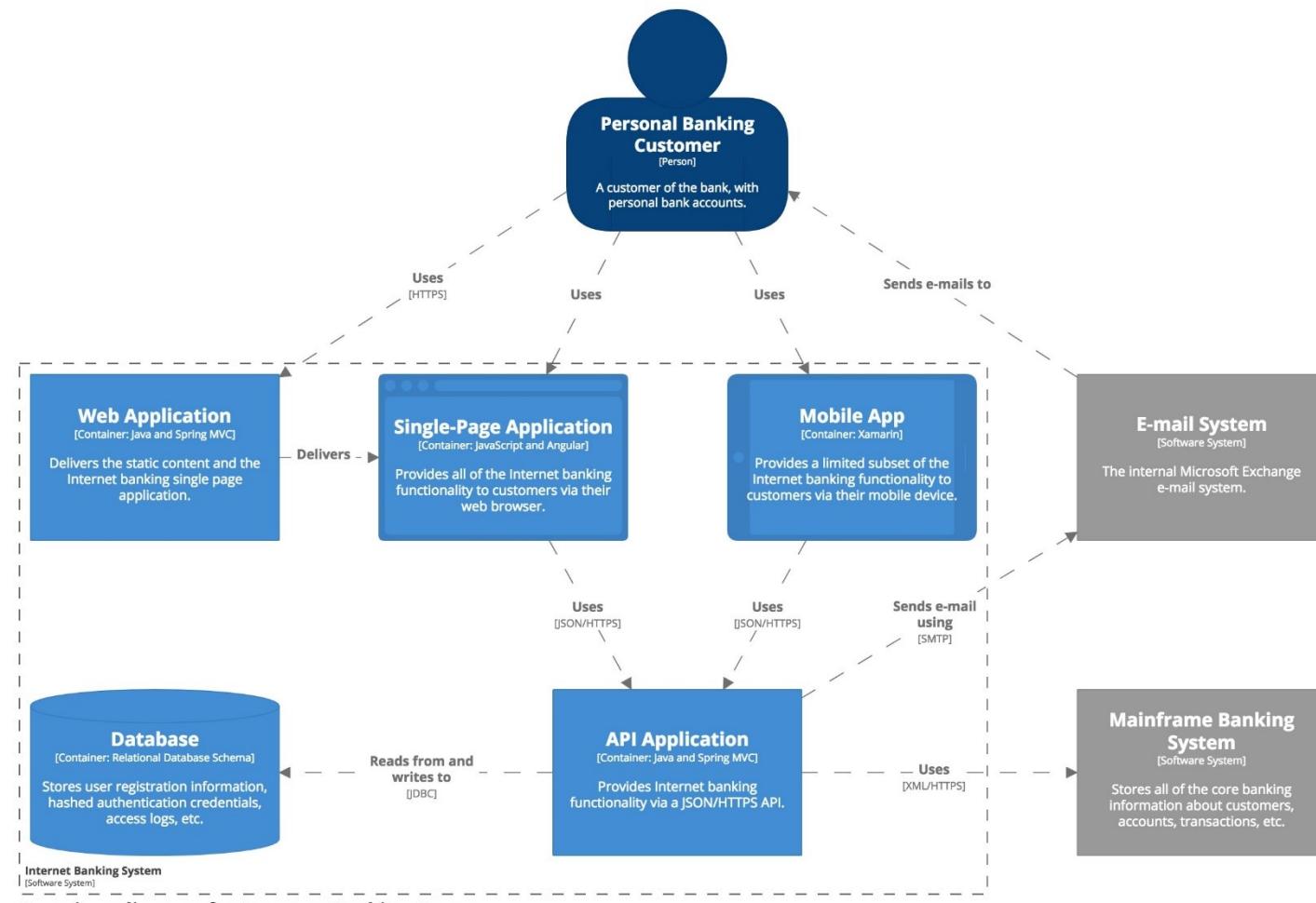
C4 Model – Level 1: System context diagram

- ▶ Shows the software system you are building
- ▶ How it fits into the world in terms of the people who use it and the other software systems it interacts with
- ▶ Colours – Systems already exist (the grey boxes) and those to be built (blue)



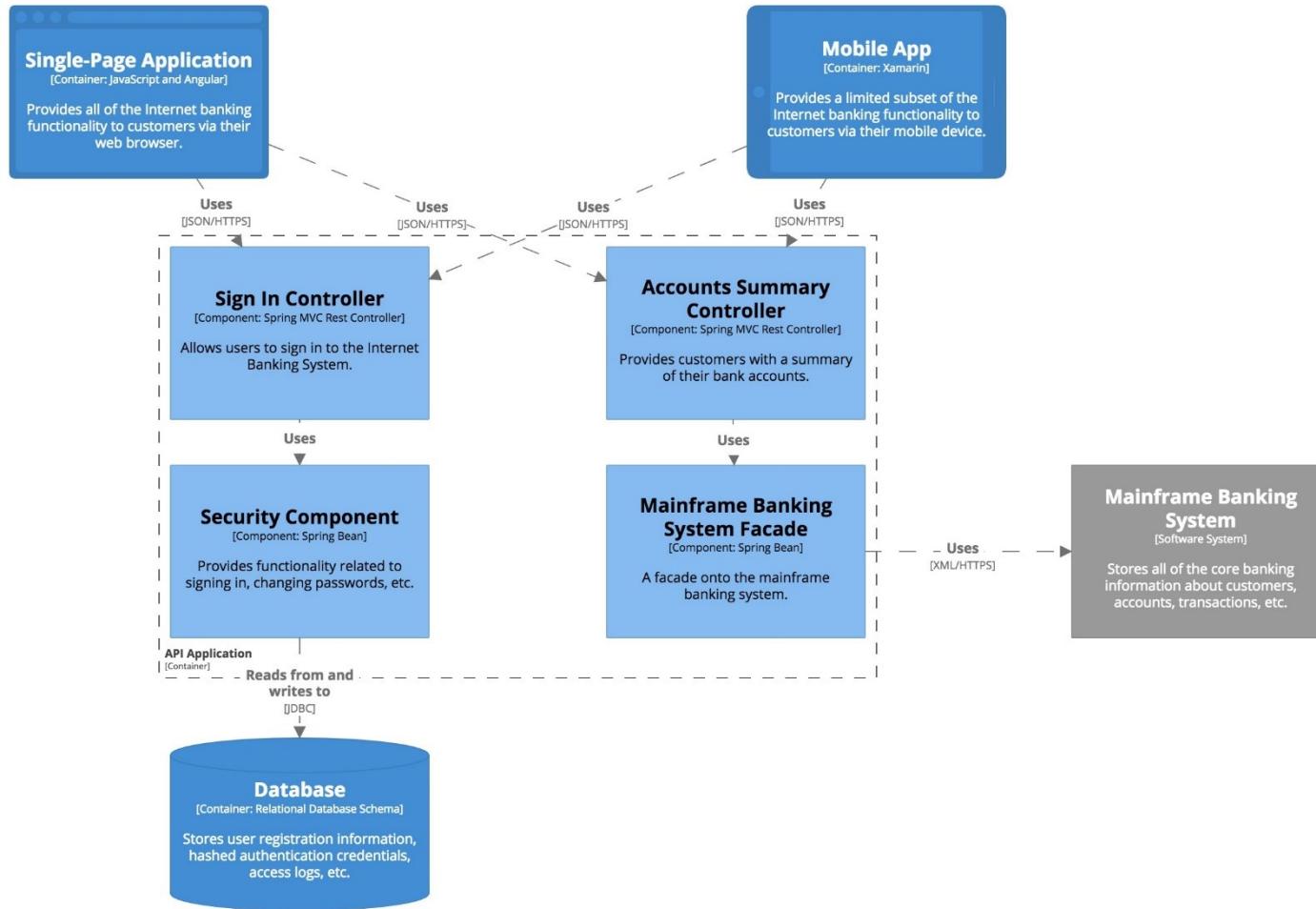
C4 Model – Level 2: Container diagram

- ▶ Zooms into the software system, and shows the containers (applications, data stores, microservices, etc.)



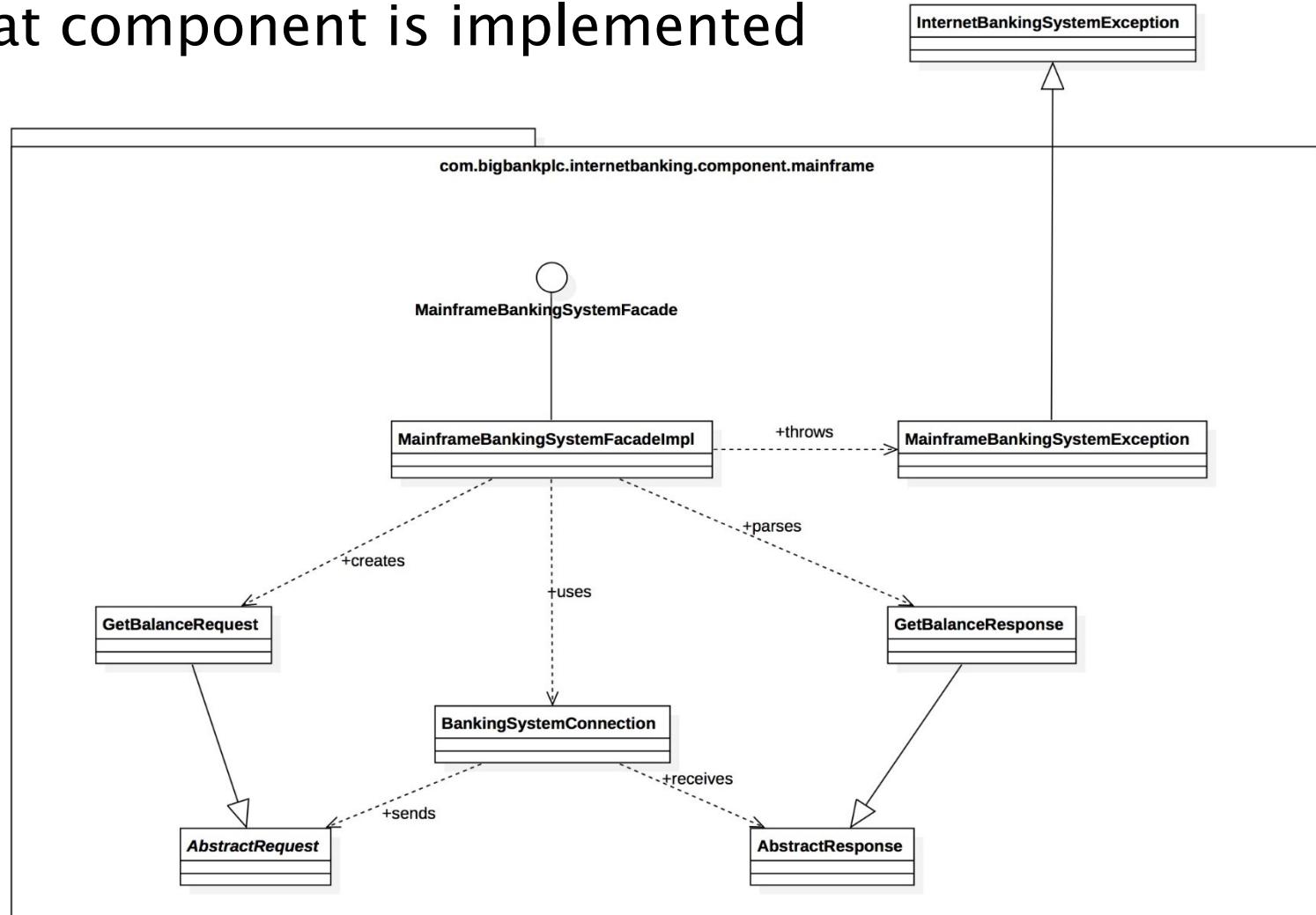
C4 Model – Level 3: Component diagram

- ▶ Zooms into an individual container to show the components inside it



C4 Model - Level 4: Code

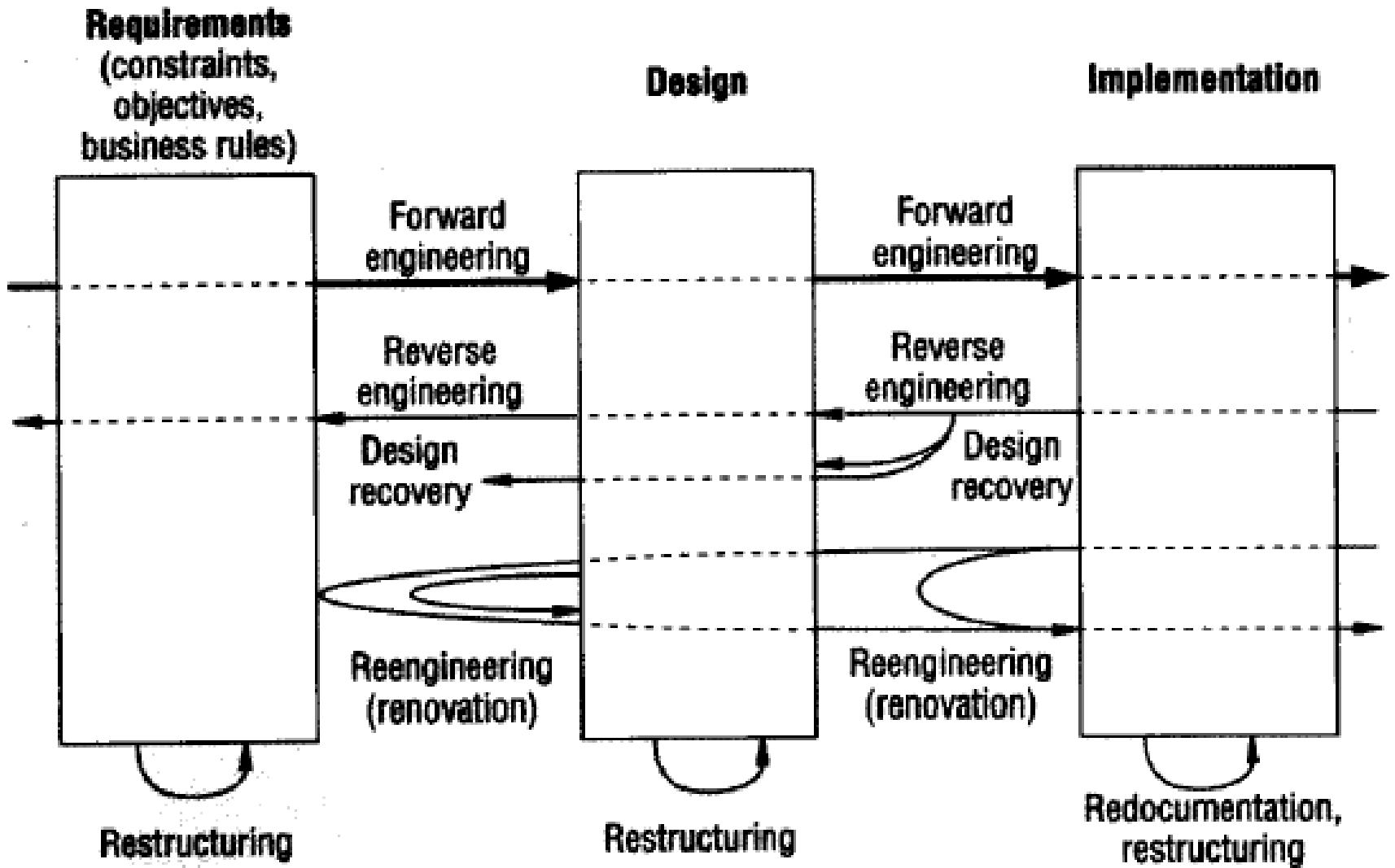
- ▶ Zoom into an individual component to show how that component is implemented



C4 Model - Links

- ▶ <https://tobiashochguertel.github.io/c4-draw.io/>
- ▶ <https://structurizr.com/express>
- ▶ <https://www.infoq.com/articles/C4-architecture-model>
- ▶ <https://c4model.com/>

Forward and Reverse Engineering



Forward Engineering

- ▶ A traditional process of moving from high-level abstractions and logical to the implementation-independent designs to the physical implementation of a system
- ▶ FE follows a sequence of going from **requirements** through **designing its implementation**

Reverse Engineering

- ▶ Reverse engineering (RE) is the process of discovering the technological principles of a device, object or system through analysis of its structure, function and operation
- ▶ *To try to make a new device or program that does the same thing without copying anything from the original*
- ▶ Reverse engineering has its origins in the analysis of hardware for commercial or military advantage

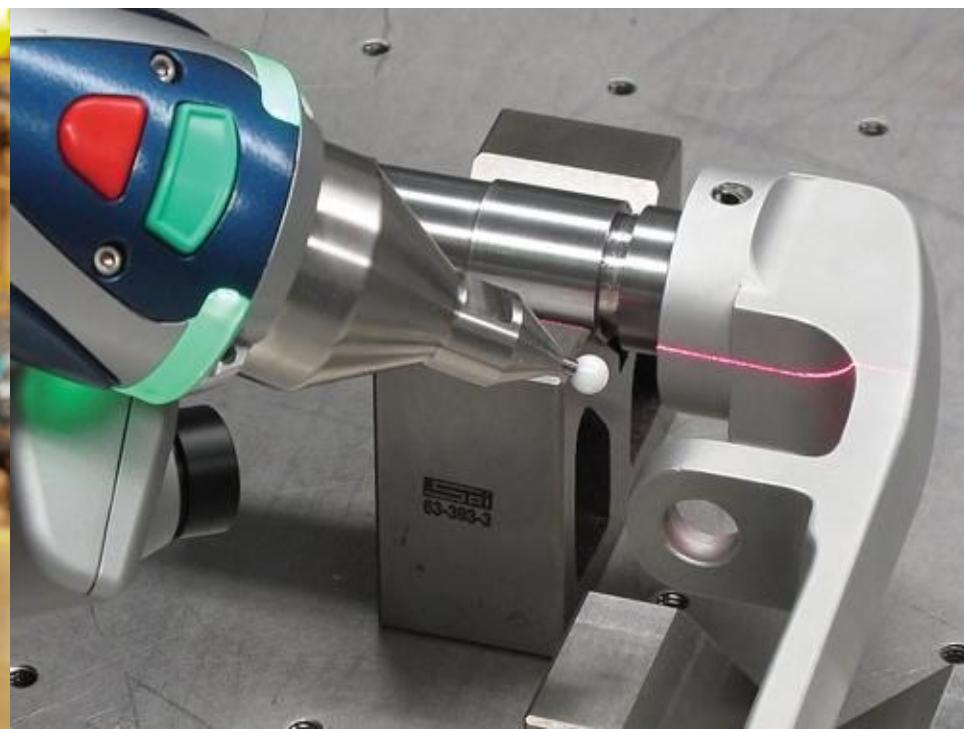
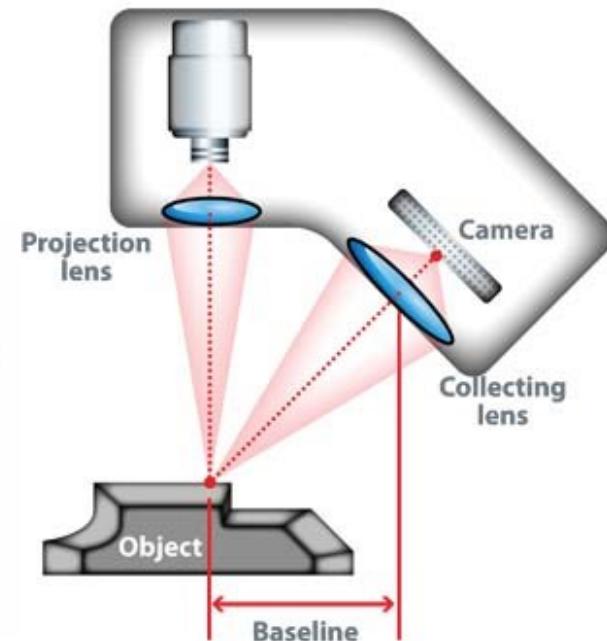
RE Motivation

- ▶ Interoperability
- ▶ Lost documentation
- ▶ Product analysis
- ▶ Security auditing
- ▶ Removal of copy protection, circumvention of access restrictions
- ▶ Creation of unlicensed/unapproved duplicates
- ▶ Academic/learning purposes
- ▶ Curiosity
- ▶ Competitive technical intelligence (understand what your competitor is actually doing versus what they say they are doing)
- ▶ Learning: Learn from others mistakes

Types of RE

- ▶ RE1: Reverse engineering of mechanical devices
- ▶ RE2: Reverse engineering of integrated circuits/smart cards
- ▶ RE3: Reverse engineering for military applications
- ▶ RE4: Reverse engineering of software

RE1: Scanere laser 3D



RE1: Servicii de modelare 3D CAD

Autodesk Design Review - C:\My Documents\Events\AU2006\Classes\AU_DataSets\Manufacturing\Shaver Complete.dwf

File Edit View Tools Help

Contents

- [5] Shaver Complete S...
- [6] Drawing - Shaver Com...
- [7] PARTS LIST
- [8] Shaver Complete.iam
- [9] Shaver Complete S...
- [10] Shaver Complete S...

Markups Properties Layers Cross Sections Model

Shaver Complete.iam

- Housing Complete Left:1
- Housing Complete Right:1
- Cover:1
- NOR-A-005:1
- Switch:1
- Shaver Complete.Harness
- NOR-P-001:1
- NOR-P-001:2
- NOR-P-001:3
- NOR-A-006:1
- NOR-P-011:1
- ISO 7045 Z M1.6x8-4.8-Z:2
- ISO 7045 Z M1.6x8-4.8-Z:1

Views

Shaver Complete.iam

Orbit around Center

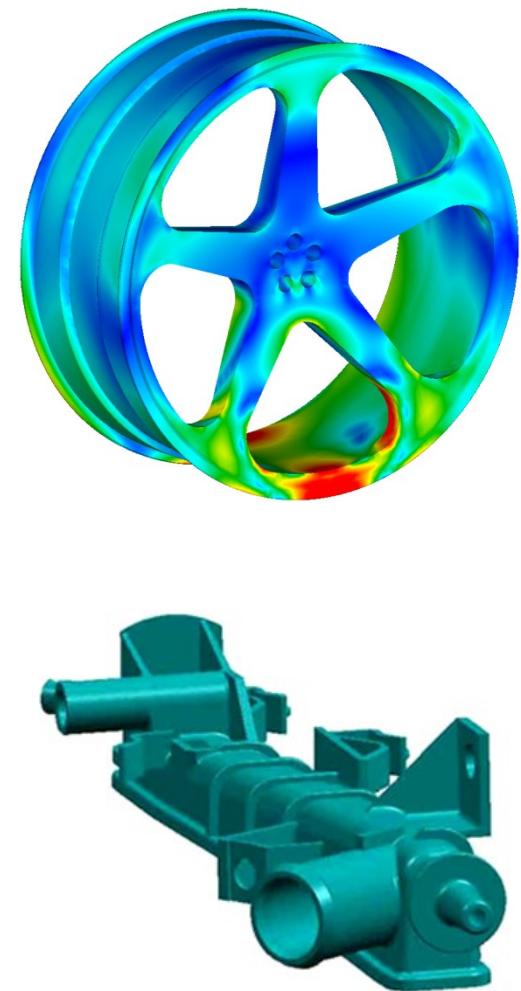
Orbit

4 of 4

Shaver Complete.iam

ITEM QTY P/N TITLE DESCRIPTION

1	1	ADSK-1101010	ELECTRONICS	PCB
3	1	ADSK-1101041	ELECTRONICS	MOTOR
4	1	ADSK-1101015	ELECTRONICS	FRONT COVER
5	3	ADSK-1101031	FORMED PART	BLADE HEAD GUARD
2	1	ADSK-1101077	ELECTRONICS	HARNESS
6	1	ADSK-1101076	MOLDED PART	HOUSING LEFT
7	2	ADSK-253001	FASTENER	PAN HEAD



RE1: Servicii de imprimare 3D

- ▶ Rapid prototyping

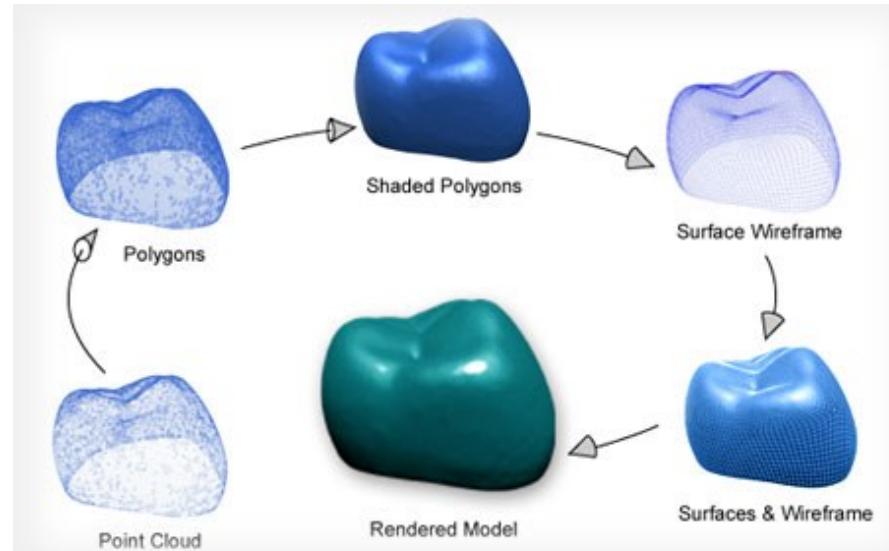


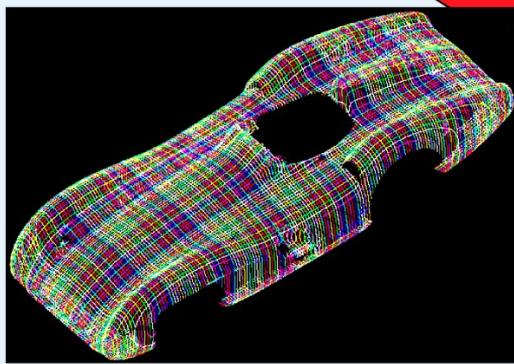
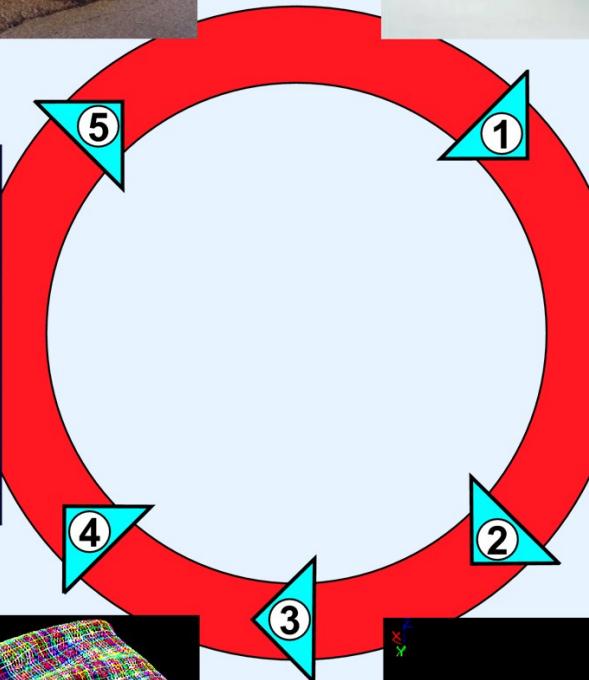
- ▶ FullCure materials



RE1: Domenii

Physical Part	Modeling Data
	
	
	

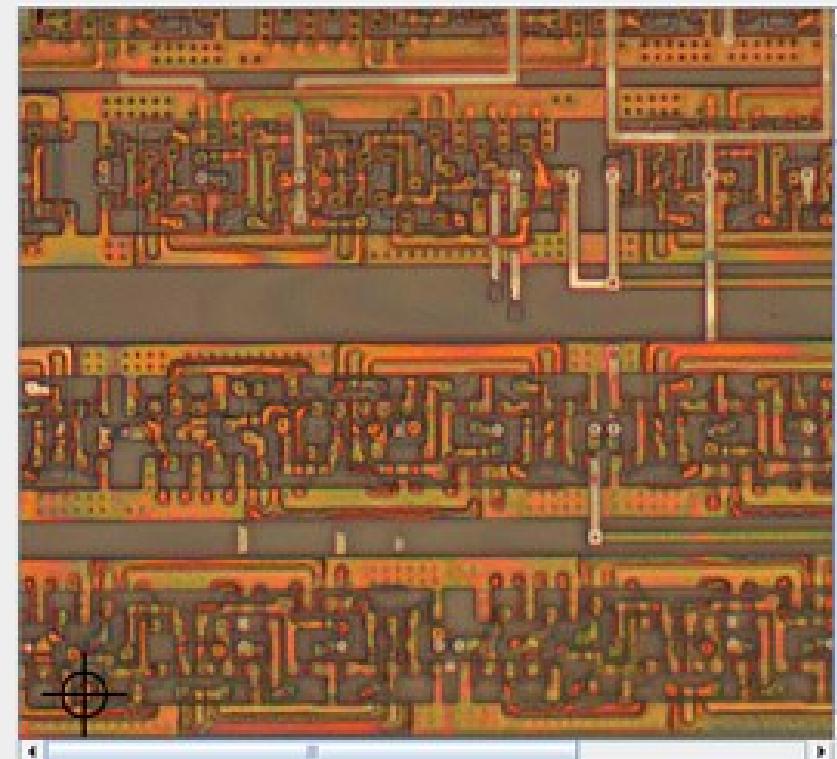
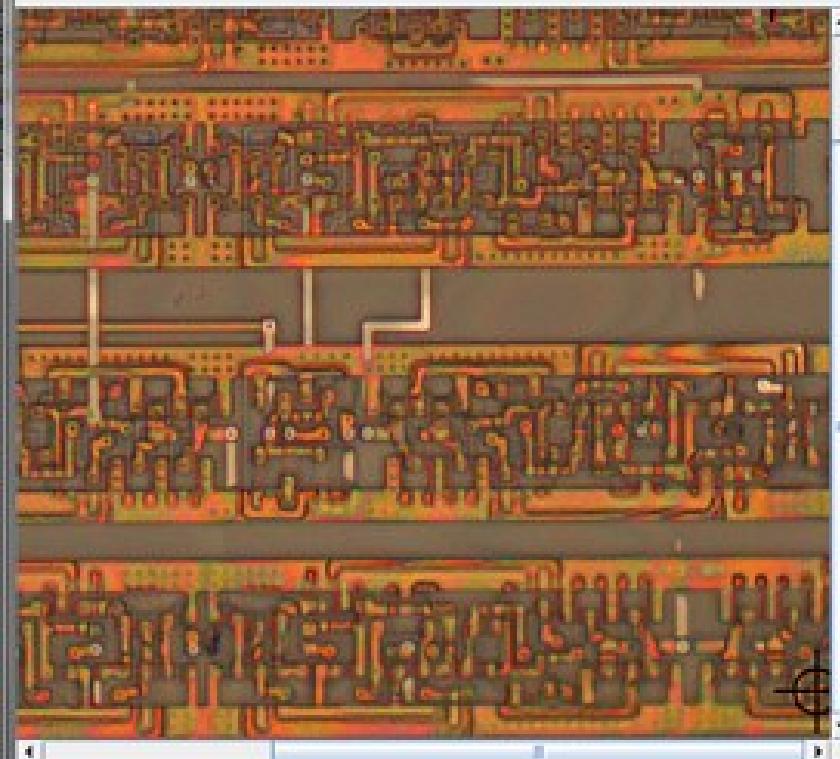




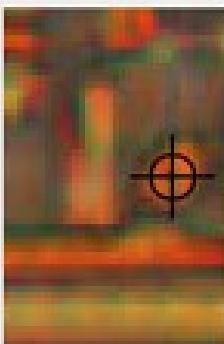
Reverse engineering of integrated circuits/smart cards

- ▶ RE is an invasive and destructive form of analyzing a smart card
- ▶ The attacker grinds away layer by layer of the smart card and takes pictures with an electron microscope
- ▶ Engineers employ sensors to detect and prevent this attack

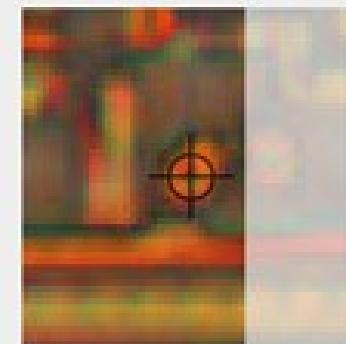
3. Schritt: Setzen der Kontrollpunkte

 D:\...\TFH\DATEIEN\...\ux\Praktikum\CREtaChip Bilder\z...

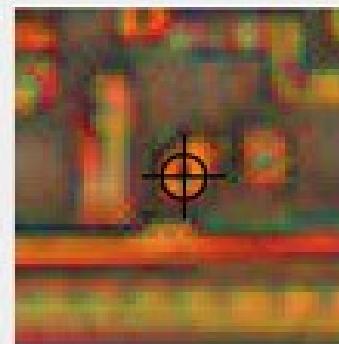
654 - 63



630 - 484

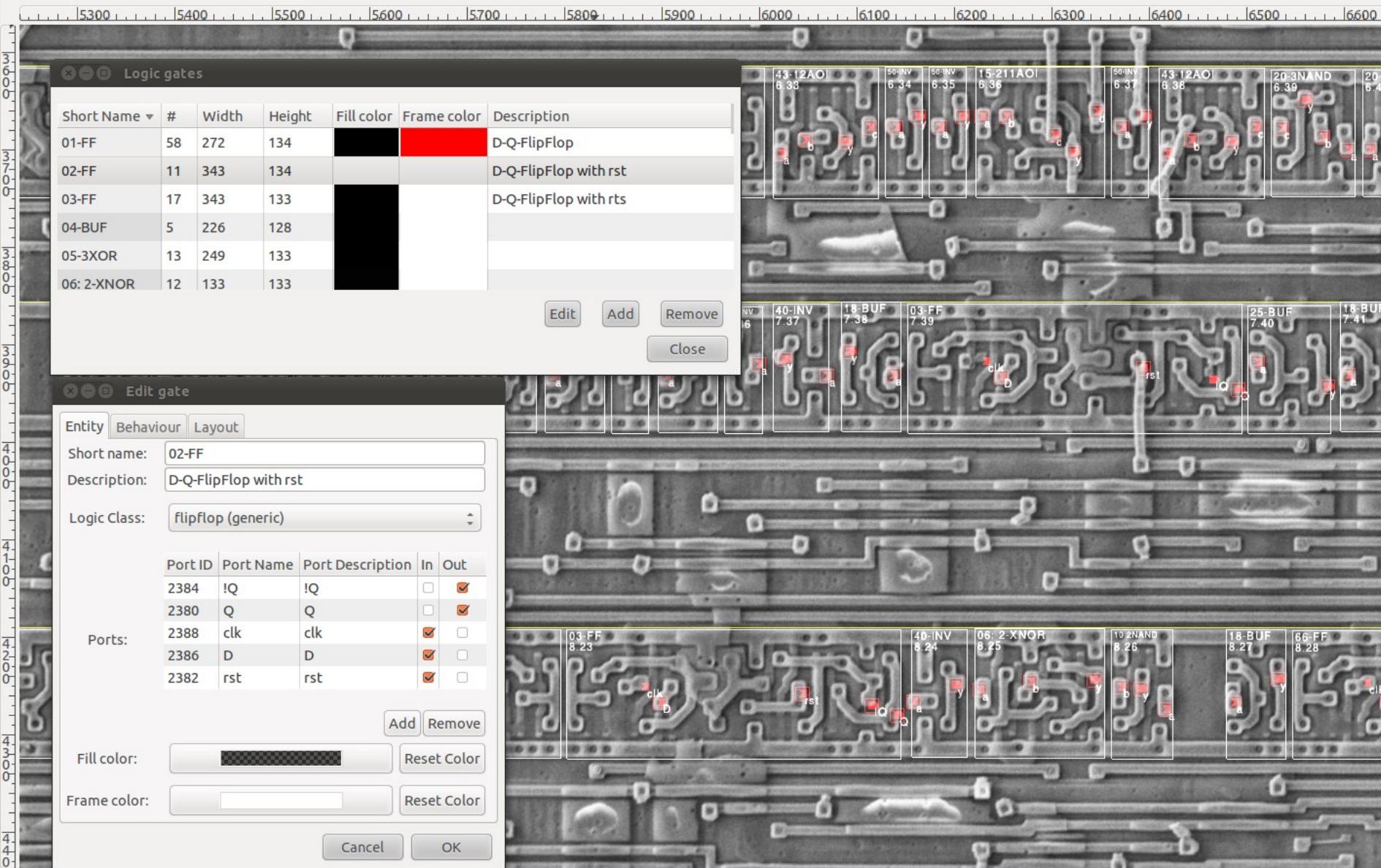


12 - 64



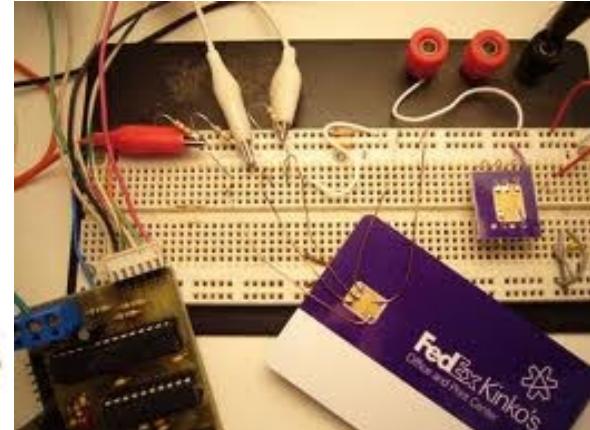
38 - 486

 Kontrollpunkt speichern • Weiter



RE2: Smart cards

- ▶ Satellite TV
- ▶ Security card
- ▶ Phone card
- ▶ Ticket card
- ▶ Bank card



Reverse engineering for military applications

- ▶ Reverse engineering is often used by militaries in order to copy other nations' technologies, devices or information that have been obtained by regular troops in the fields or by intelligence operations
- ▶ It was often used during the Second World War and the Cold War
- ▶ Well-known examples from WWII and later include: rocket, missile, bombers, China has reversed many examples of US and Russian hardware, from fighter aircraft to missiles and HMMWV cars

RE3: Avioane

► US - B-29

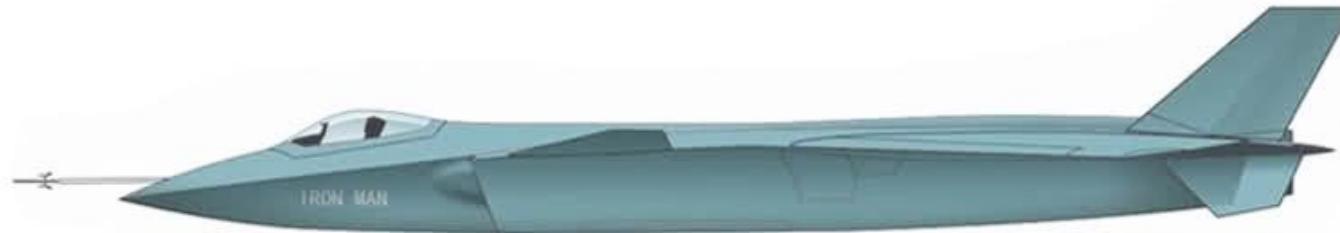


URSS - Tupolev Tu-4



RE3: Avioane (2)

- ▶ Chinese J-20, Black Eagle US F-22, Russian Sukhoi T-50



RE3: Rachete

► US -AIM-9 Sidewinder

Soviet – Vympel K-13



Russia's new ballistic missile submarine

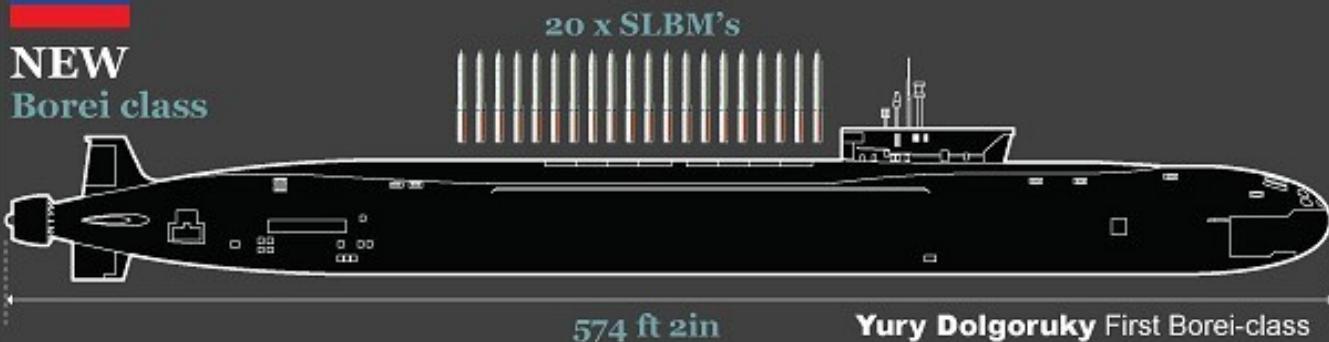
RE3: Submarines



OLD
Typhoon class



NEW
Borei class



Yury Dolgoruky First Borei-class submarine is undergoing sea trials, two others are being built



Royal Navy
Vanguard class



RE3: UFOs

United States Patent [191]

3,774,865

Patent

Nov. 27, 1973

[192] FLYING SAUCER

p. 540.

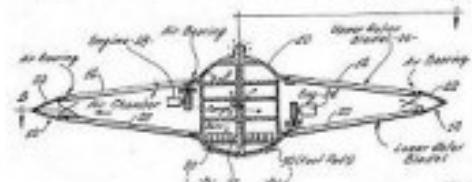
[194] Inventor: Olympe F. Ribe, Rue Vicente da
Graça Palmeira, 61, Rio De Janeiro,
Brazil

[195] Filed: Jan. 5, 1972

[196] Appl. No.: 214,623

Related U.S. Application Data

[197] Continuation-in-part of Ser. No. 40,536, May 16,



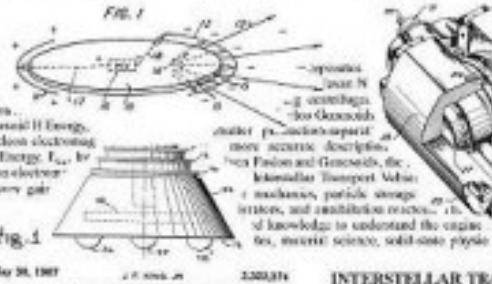
Feb. 20, 1962

T. T. BROWN
ELECTROSTATIC GENERATOR

3,022,4

8 Sheets-Shaut 1

Patent Exp. 5, 1987



May 26, 1967

2,202,374

INTERSTELLAR TRA

3,625,605

ABSTRACT

A flying saucer type of aircraft or water vehicle is provided, which may take the form of a toy, or of an actual full-sized passenger and cargo carrying vehicle.

It has a circular shaped

section and an inner

group of outer heli-

copter blades

at the central point

and the blades

are mounted on four

rotatably driven by the

helicopter blades de-

velopment. Related

which may be identi-

cate of the vehicle, or

of the direction of

W. L. S.

PLATE PREP.

Patent Exp.

6 Sheets-Sheet 1

Reverse Engineering a UFO

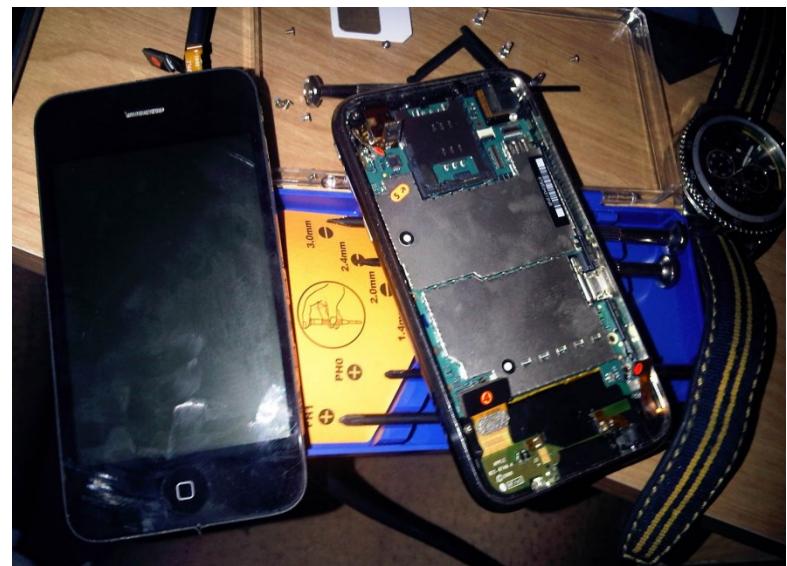


by Robert Klein

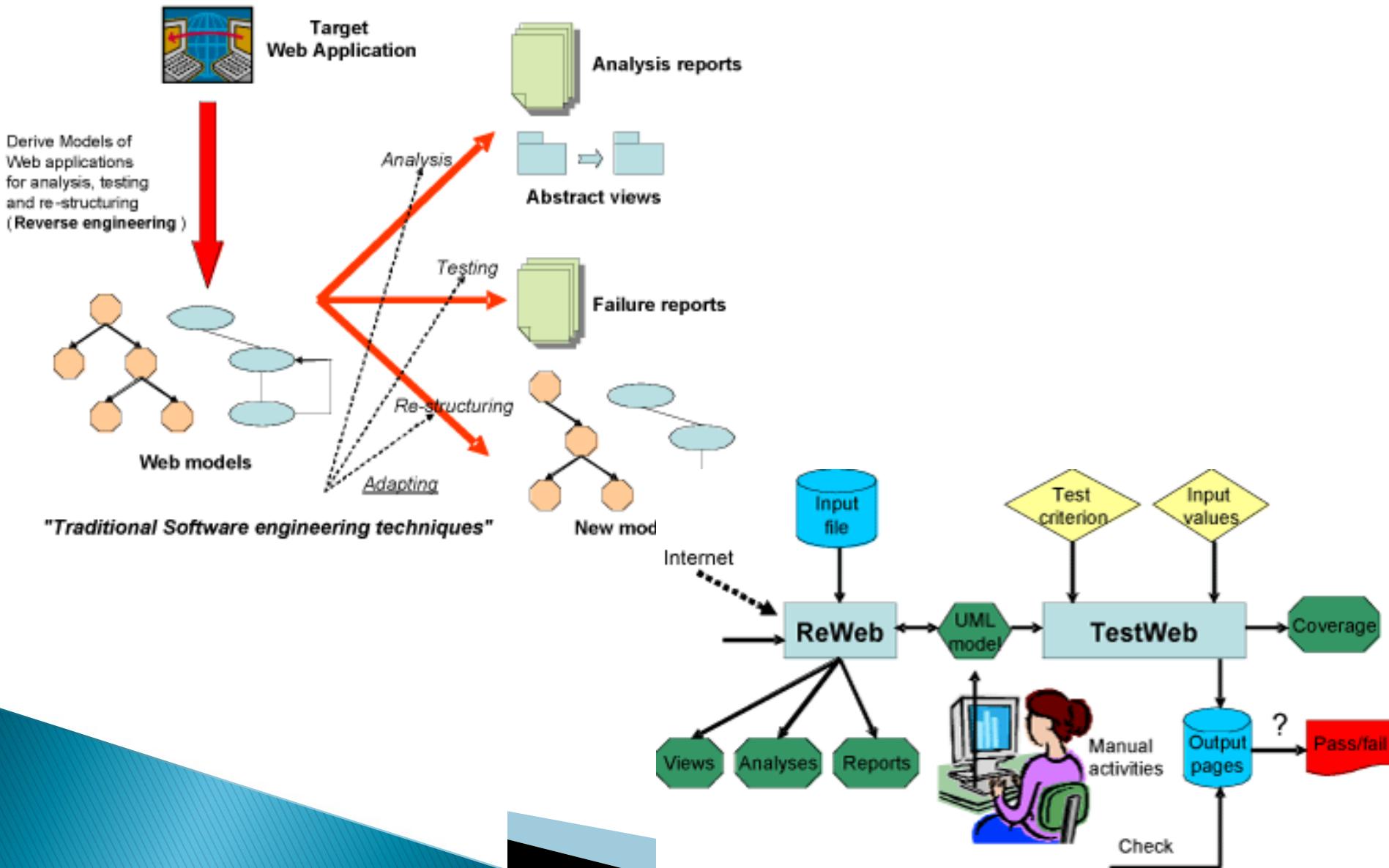
Reverse engineering of software

- ▶ Reverse engineering is the process of analyzing a subject system to create representations of the system at a higher level of abstraction
- ▶ In practice, two main types of RE emerge:
 - Source code is available (but it is poorly documented)
 - There is no source code available for the software
- ▶ **Black box testing** in software engineering has a lot in common with reverse engineering

RE4: Smart phones



RE4 of Web Applications



RE4: DJ Java Decompiler

```
public class Test
{
    private int n;
    private int m;

    public static void main(String
args[])
    {
        for(int i=1;i<10;i++)
            System.out.println("Test");
    }
}
```

The screenshot shows the DJ Java Decompiler application window titled "Test.jad - DJ Java Decompiler". The interface includes a menu bar with File, Edit, Search, View, Settings, Language, Tools, and Help. Below the menu is a toolbar with various icons for file operations like Open, Save, Print, and Cut/Paste. The main workspace displays the decompiled Java code for the "Test" class. The code is color-coded: red for keywords like import, public, static, void, and for loops; green for strings; and black for comments and variable names. The code itself is:

```
import java.io.PrintStream;

public class Test
{
    public Test()
    {
    }

    public static void main(String args[])
    {
        for(int i = 1; i < 10; i++)
            System.out.println("Test");
    }

    private int n;
    private int m;
}

public Test() {}
public static void main(String args[]) {}
```

At the bottom of the window, status bars show "Line: 21 Col: 1", "Modified", "Num lock: OFF", "Caps lock: OFF", and "Ir". On the right side of the window, there is a vertical toolbar with various icons representing different tools or features of the decompiler.

RE4: JAD

- ▶ Link: <http://www.steike.com/code/java-reverse-engineering/>
- ▶ jad.exe NumeFisier.class => NumeFisier.jad

D:\Work\Ady\Scoala\2009-2010\IP\Cursuri\04\Reverse\Initial\Student.java - Notepad++

```
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?  
Student.java  
1  /*  
2   * To change this template, choose Tools | Templates  
3   * and open the template in the editor.  
4   */  
5  
6 package test1;  
7  
8 /**  
9  *  
10 * @author Adrian  
11 */  
12 public class Student extends Person{  
13     private int year;  
14     private int group;  
15  
16     public Student(String name, String address,int year, int gro  
17 {  
18         super(name,address);  
19         this.group = group;  
20         this.year = year;  
21     }  
22 }  
23
```

D:\Work\Ady\Scoala\2009-2010\IP\Cursuri\04\Reverse\Student.jad - Notepad++

```
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?  
Student.jad  
1  // Decompiled by Jad v1.5.8f. Copyright 2001 Pavel Kouznetsov.  
2  // Jad home page: http://www.kpdus.com/jad.html  
3  // Decompiler options: packimports(3)  
4  // Source File Name: Student.java  
5  
6 package test1;  
7  
8 // Referenced classes of package test1:  
9 //          Person  
10  
11 public class Student extends Person  
12 {  
13  
14     public Student(String name, String address, int year, int group)  
15     {  
16         super(name, address);  
17         this.group = group;  
18         this.year = year;  
19     }  
20  
21     private int year;  
22     private int group;  
23 }  
24  
25
```

RE4: Open Office

ss - OpenOffice Untitled1 - OpenOffice.org Impress

File Edit View Insert Tools Slide Show Window Help

File Edit View Insert Format Tools Slide Show Window Help

Normal Outline Notes Handout Slide Sorter

Tasks View

Master Pages

Layouts

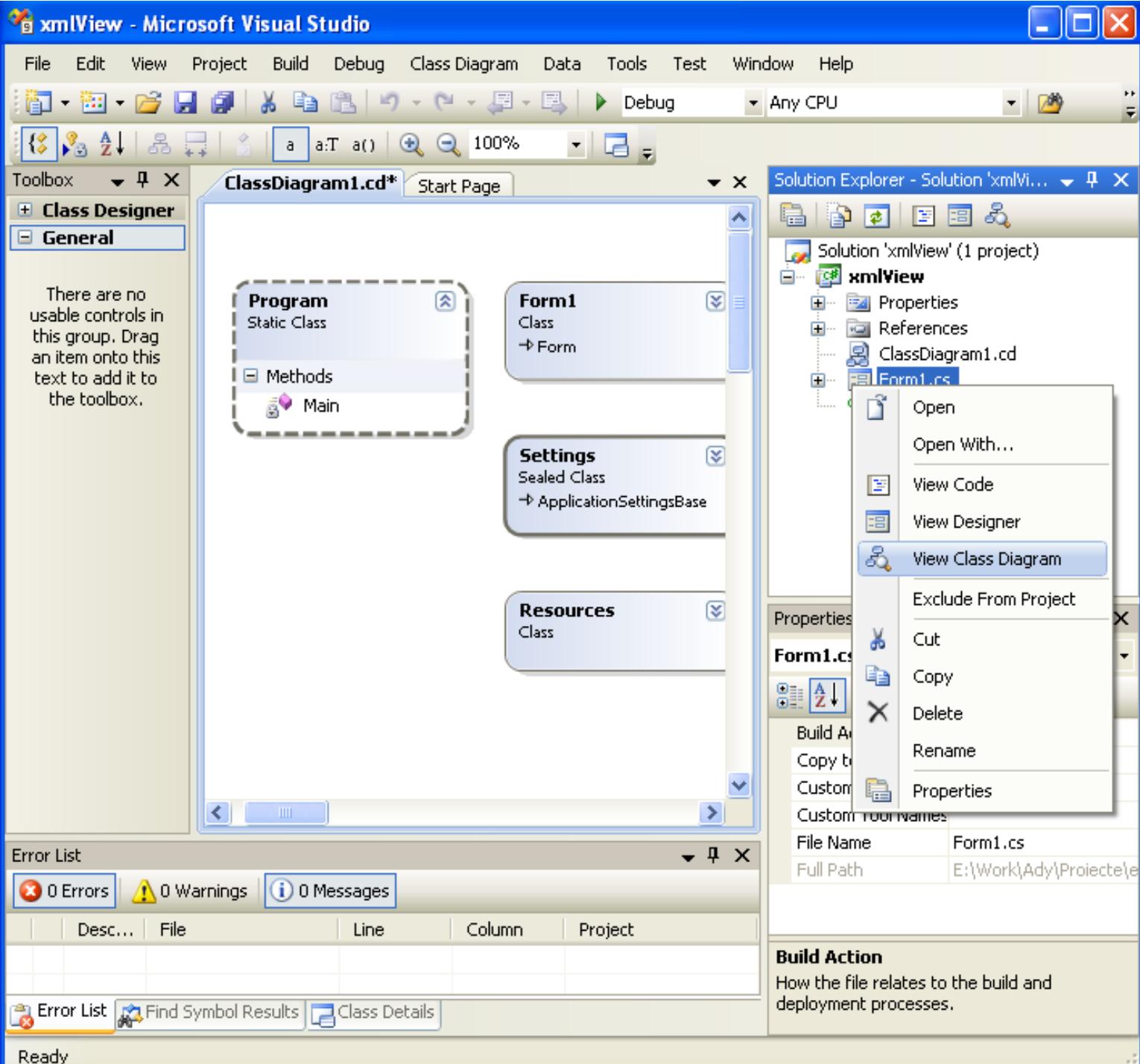
Exemplu OpenOffice.org

- Alt exemplu
- și altul
- și încă unul
- și încă unul

29,16 / -2,59 0,00 x 0,00 51% * Slide 1 / 1 Default

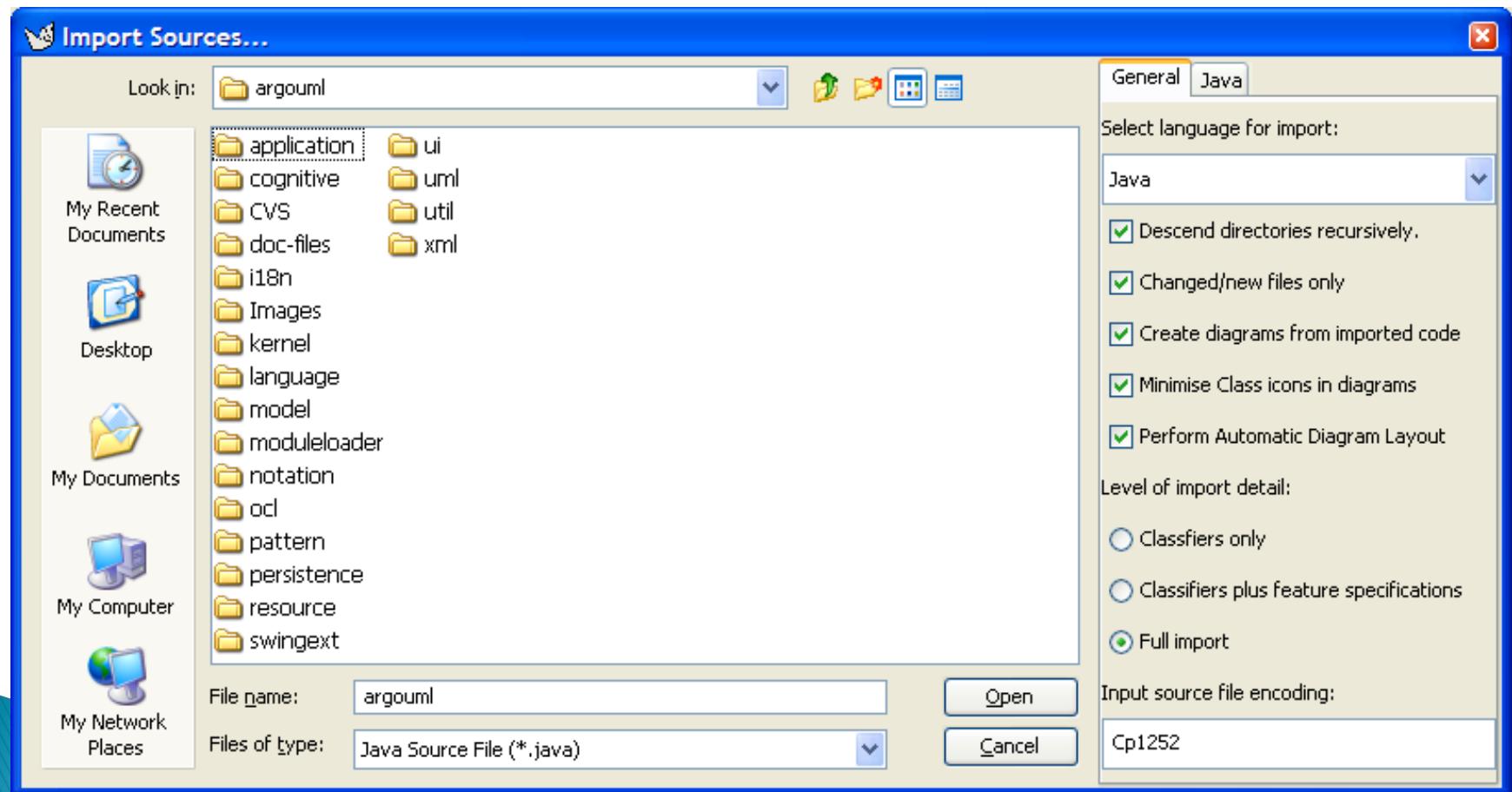
The screenshot shows the OpenOffice.org Impress application interface. On the left, there is a Calc spreadsheet window titled 'ss - OpenOffice' showing a single sheet with data from K23 to K35. The main Impress window has a slide titled 'Exemplu OpenOffice.org' with four bulleted points: 'Alt exemplu', 'și altul', 'și încă unul', and 'și încă unul'. To the right of the text is a pie chart divided into four segments: blue, green, yellow, and red. Below the pie chart are three yellow stars. The Impress toolbar at the bottom includes icons for selection, text, shapes, and other slide editing tools.

C#

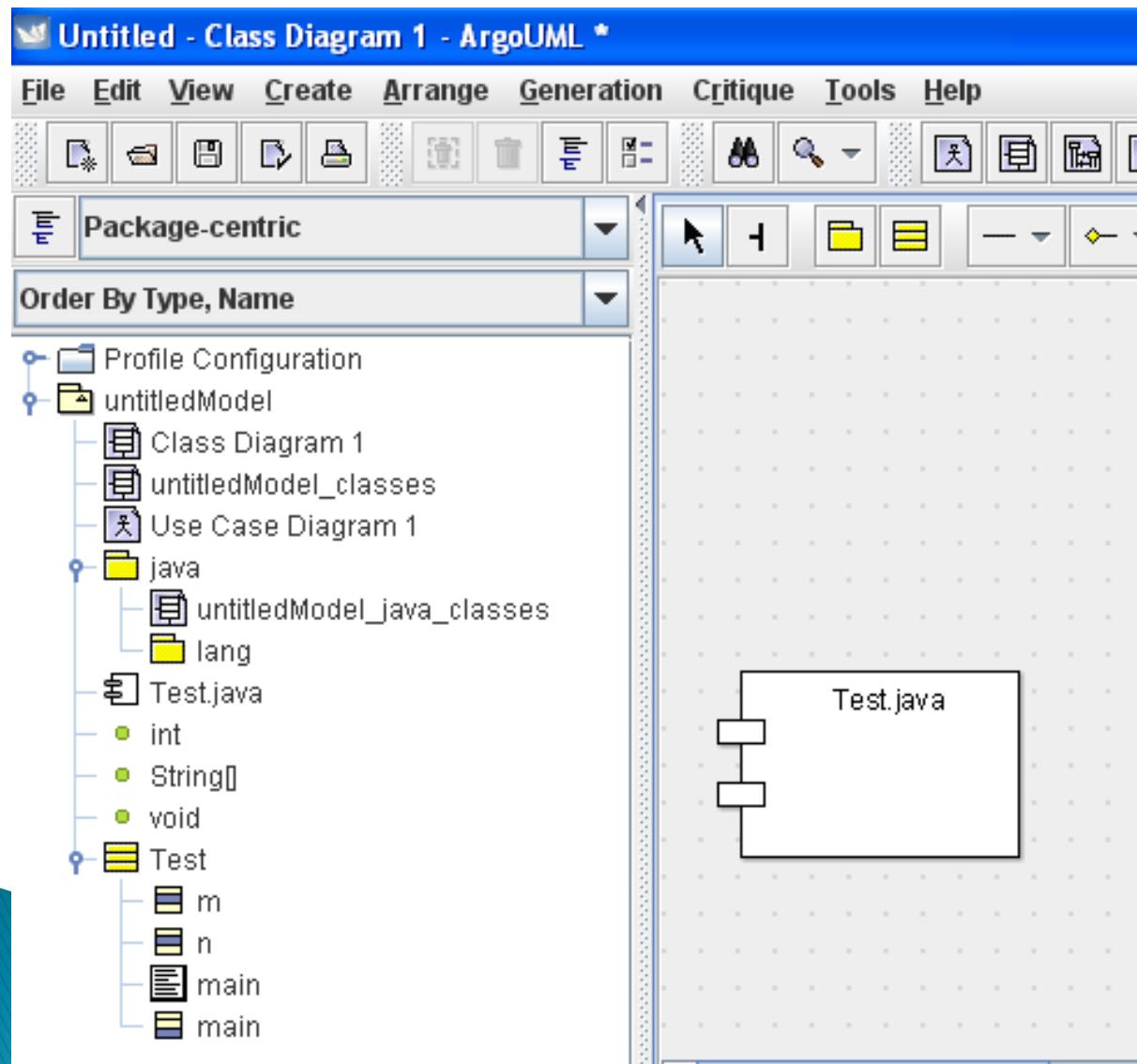


RE în ArgoUML

- ▶ File -> Import Sources...



Pentru exemplul anterior...



Demo FE & RE

- ▶ Forward engineering:
 - Diagramme de clasă -> .java files (ArgoUML)
 - .java files -> .class files (NetBeans)

- ▶ Reverse engineering:
 - .class files -> .java files (JAD Decompiler)
 - .java files -> Diagramme de Clasă (ArgoUML)

Concluzii

- ▶ Diagrame UML:
 - Interacțiuni
 - Comportamentale
 - Structură
- ▶ C4 Model: context, container, component, code

Întrebări

- ▶ 1) Până unde trebuie să modelați un proiect la care lucrați? (un proiect cu 4–5 persoane)
- ▶ 2) Cu ce ar trebui începută modelarea unui proiect?
- ▶ 3) E etic ca o firmă (mare) să facă RE pe un produs de la concurență?
- ▶ 4) De ce nu trebuie încurajat RE? Care sunt efectele RE?

Bibliografie

- ▶ Ovidiu Gheorghieş, Curs 5 IP
- ▶ www.uml.org
- ▶ Reverse Engineering and Design Discovery: A Taxonomy, Chikofsky, E.J. and Cross, J., January, 1990

Links (RE)

- ▶ **DJ Java Decompiler 3.10.10.93:**
<http://www.softpedia.com/progDownload/DJ-Java-Decompiler-Download-13481.html>
- ▶ **Open Office:** <http://ro.wikipedia.org/wiki/OpenOffice.org>
- ▶ **UML Reverse Engineering for Existing Java, C# , and Visual Basic .NET Code:**
<http://www.altova.com/umodel/uml-reverse-engineering.html>
- ▶ **Reverse Engineering:**
http://en.wikipedia.org/wiki/Reverse_engineering
- ▶ **PROTO 3000 3D Engineering Solutions:**
<http://www.proto3000.com/services.aspx>
- ▶ **HAR2009:** <http://www.degate.org/HAR2009/>
- ▶ **Degate:** <http://www.degate.org/screenshots/>
- ▶ **Inteligent:** <http://www.intelligentrd.com/>
- ▶ **Smartphones RE:** <http://www.cytraxsolutions.com/2011/01/smartphones-security-and-reverse.html>