

Ingineria Programării

Cursul 4 – 10 Martie
adiftene@infoiasi.ro

Cuprins

- ▶ Din Cursurile trecute...
- ▶ Forward Engineering
- ▶ Reverse Engineering
- ▶ Diagrame de Interacțiuni
 - Diagrame de Secvență
 - Diagrame de Colaborare

Din cursurile trecute...

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

Elm327



bluetooth

Elm327



Elm327

bluetooth

SMS

SMS

SMS

Elm327

Elm327

Elm327

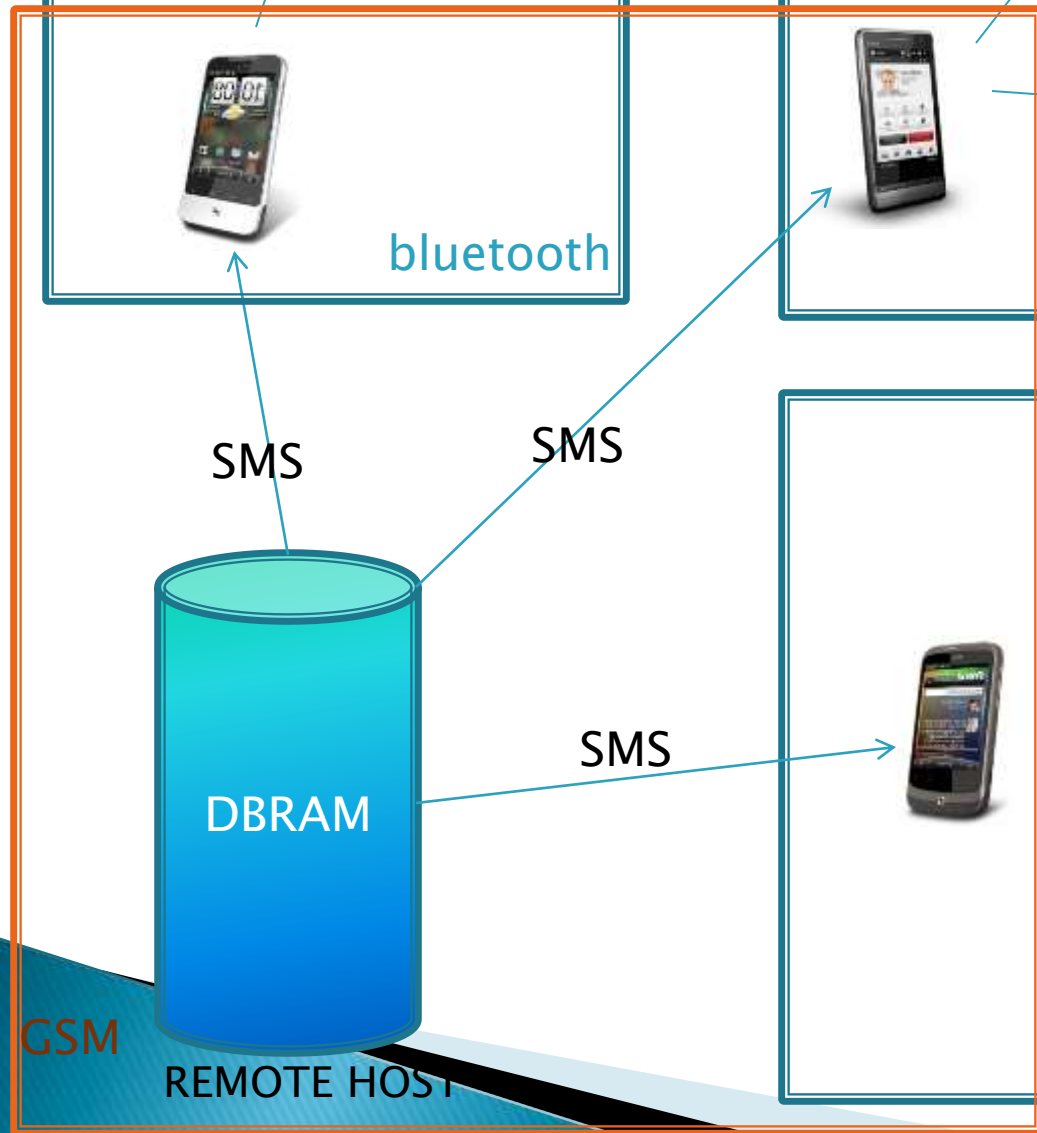
bluetooth

DBRAM

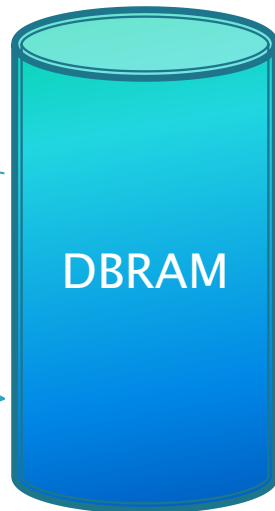


GSM

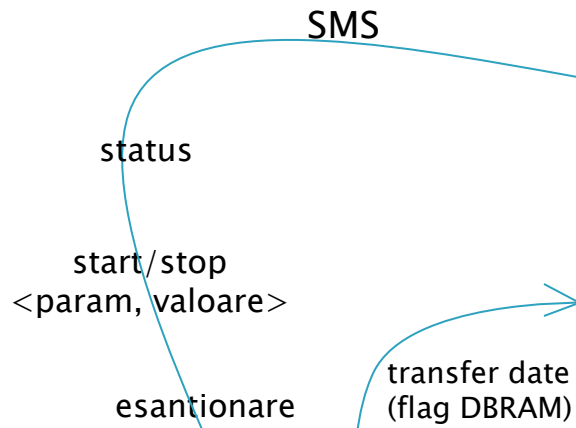
REMOTE HOST



REMOTE HOST



gestionare
prelucrare
interpretare
vizualizare
monitorizare
atentionare



data
ora



telefon
mobil
(IMEI, GPS)

colectare date

start
configurare

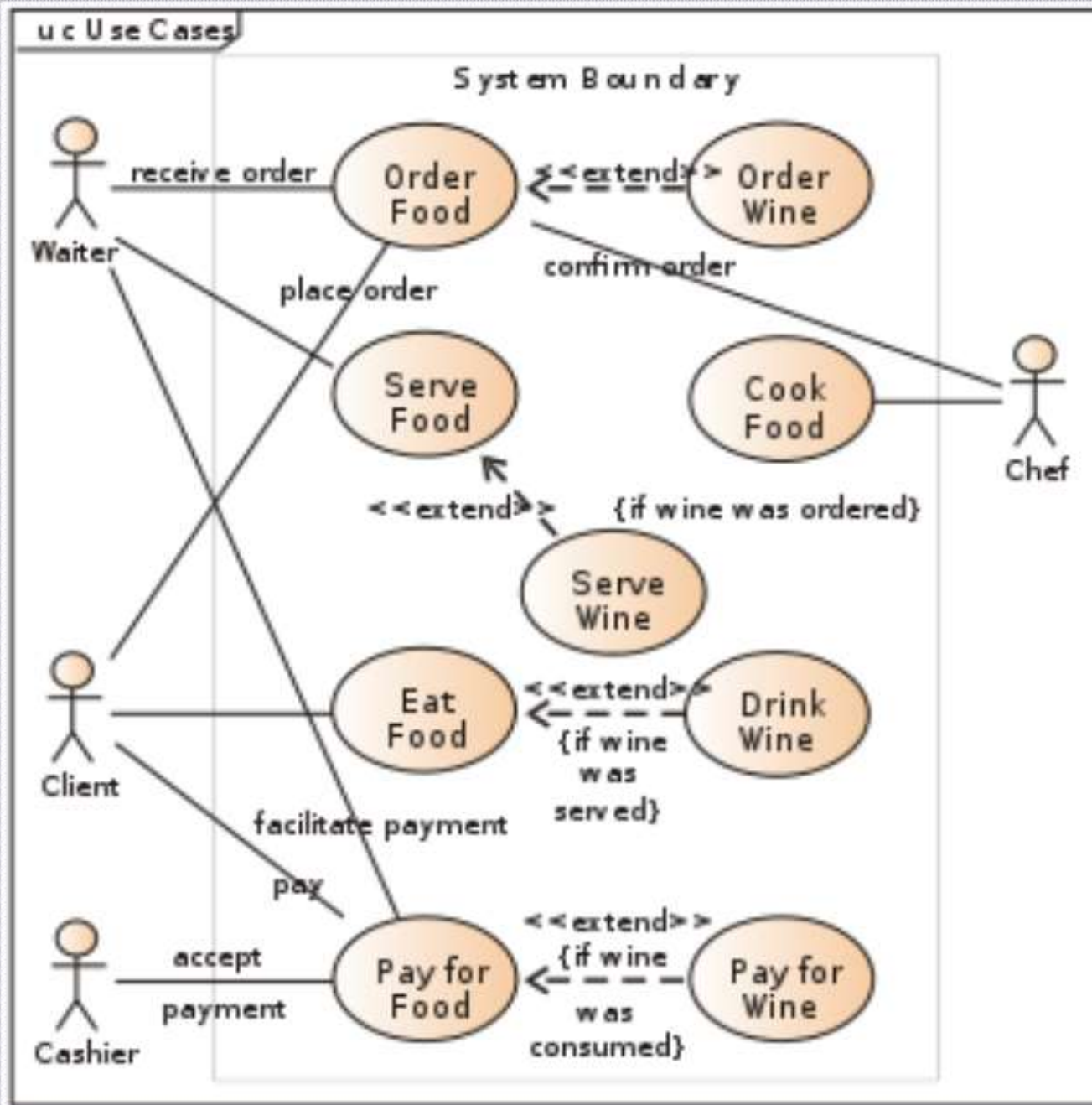
Elm327

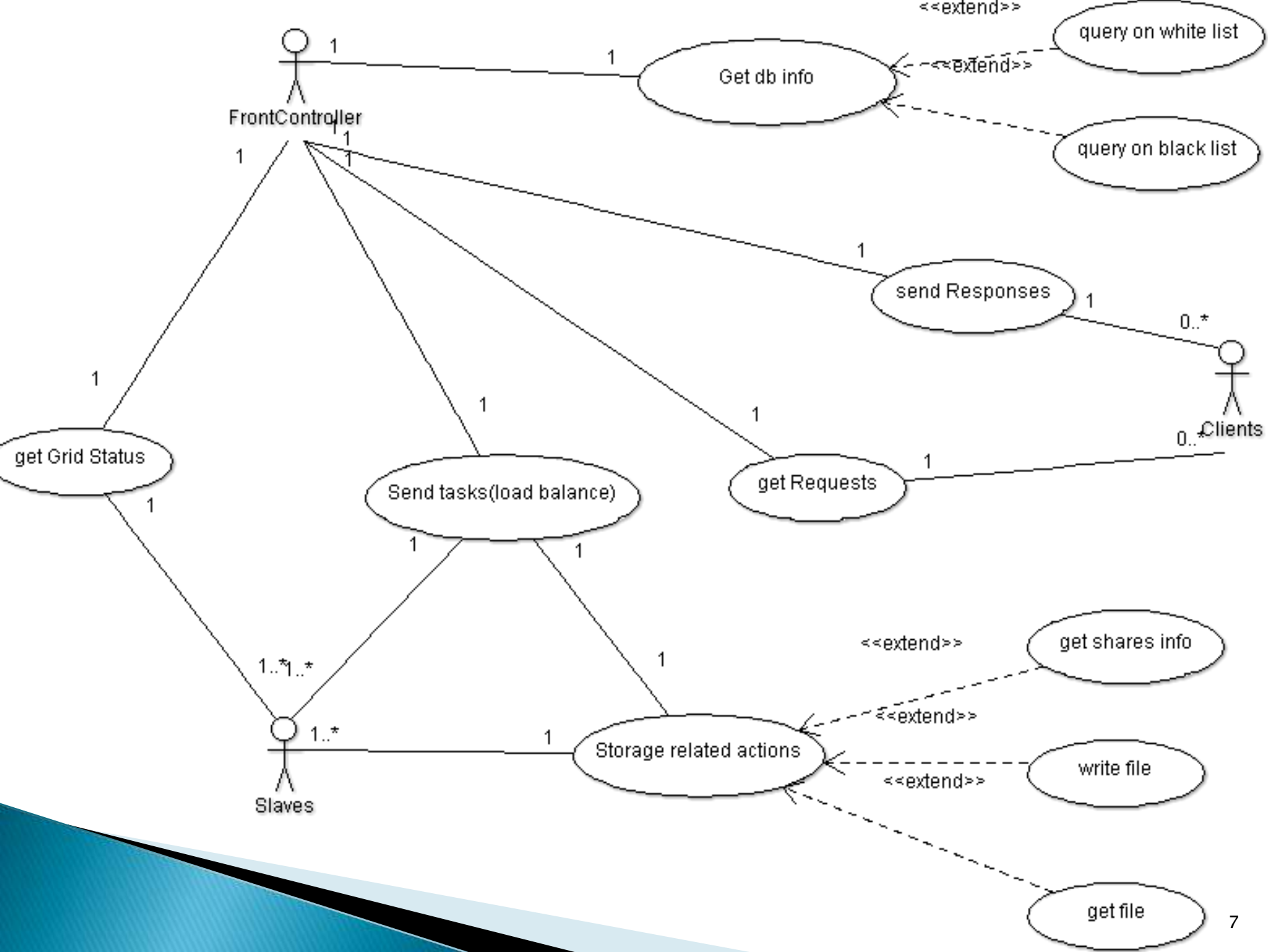
stop
configurare

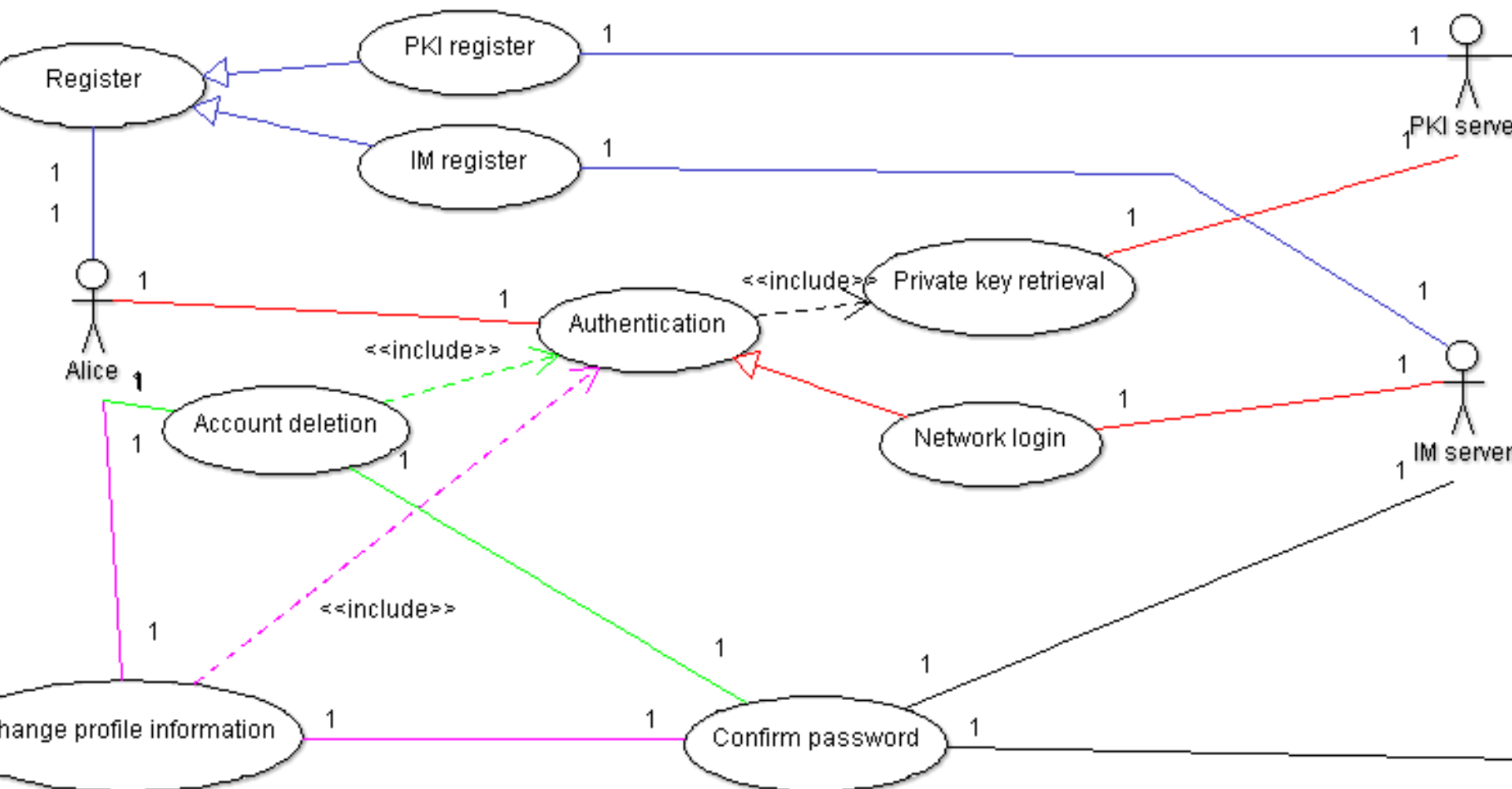
colectare date

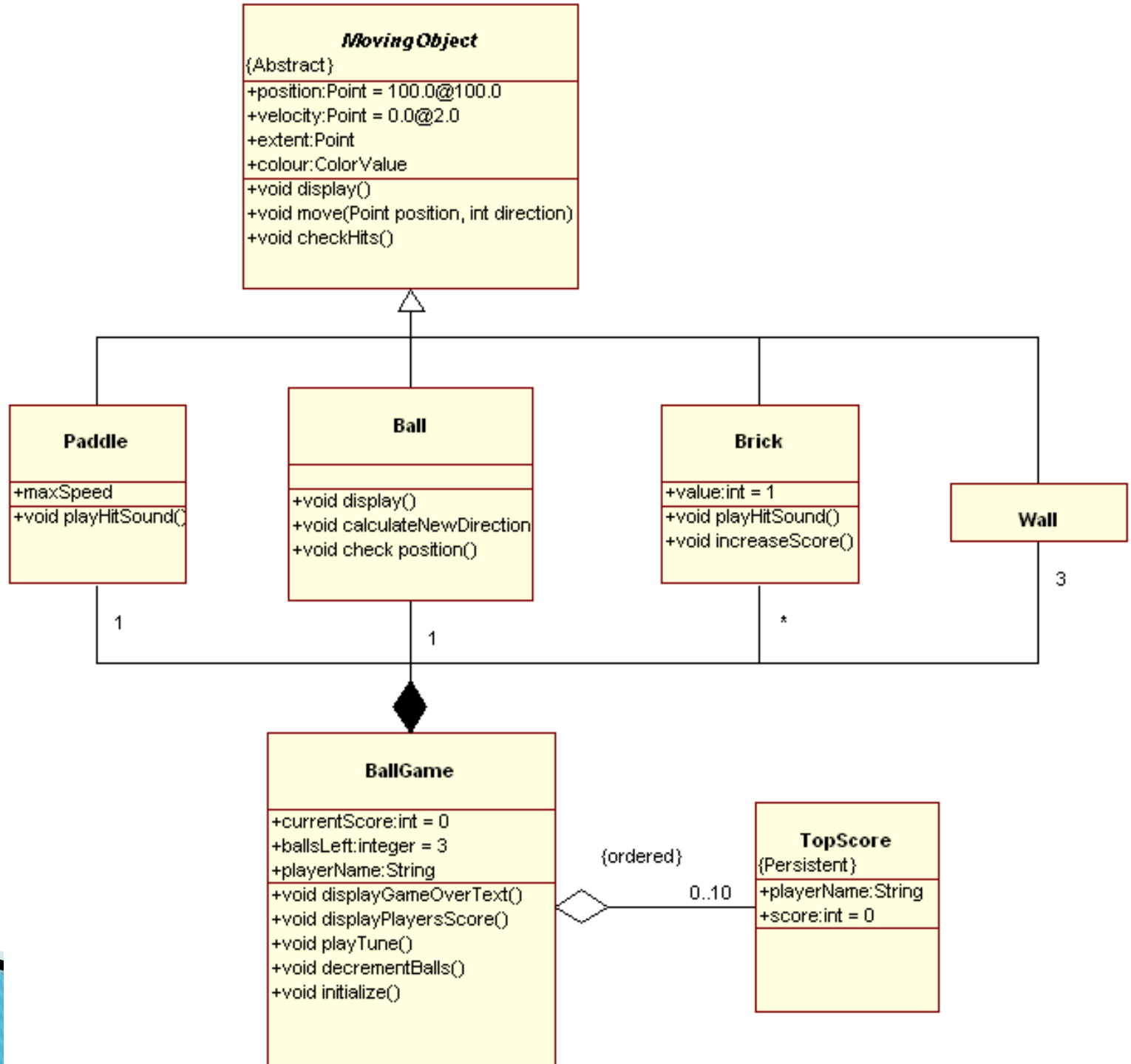
Elm327

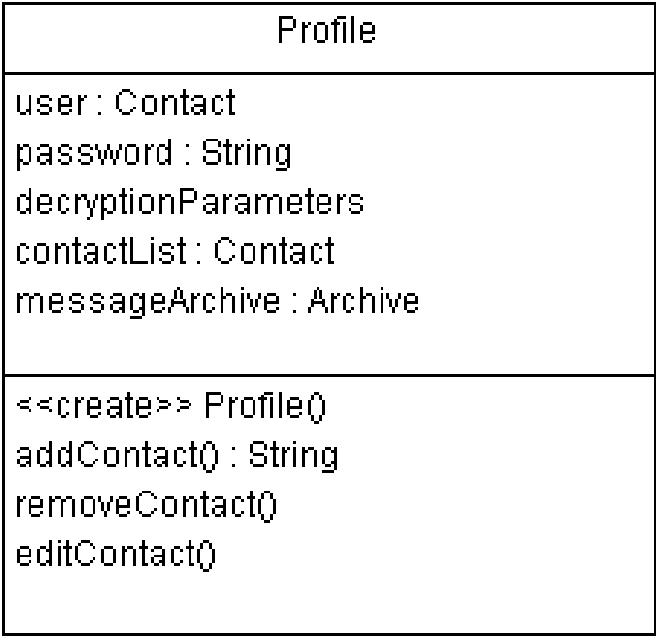
Echipamente





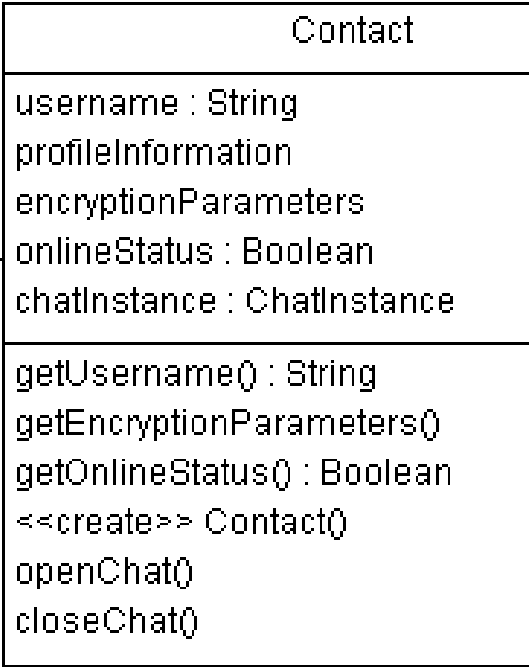




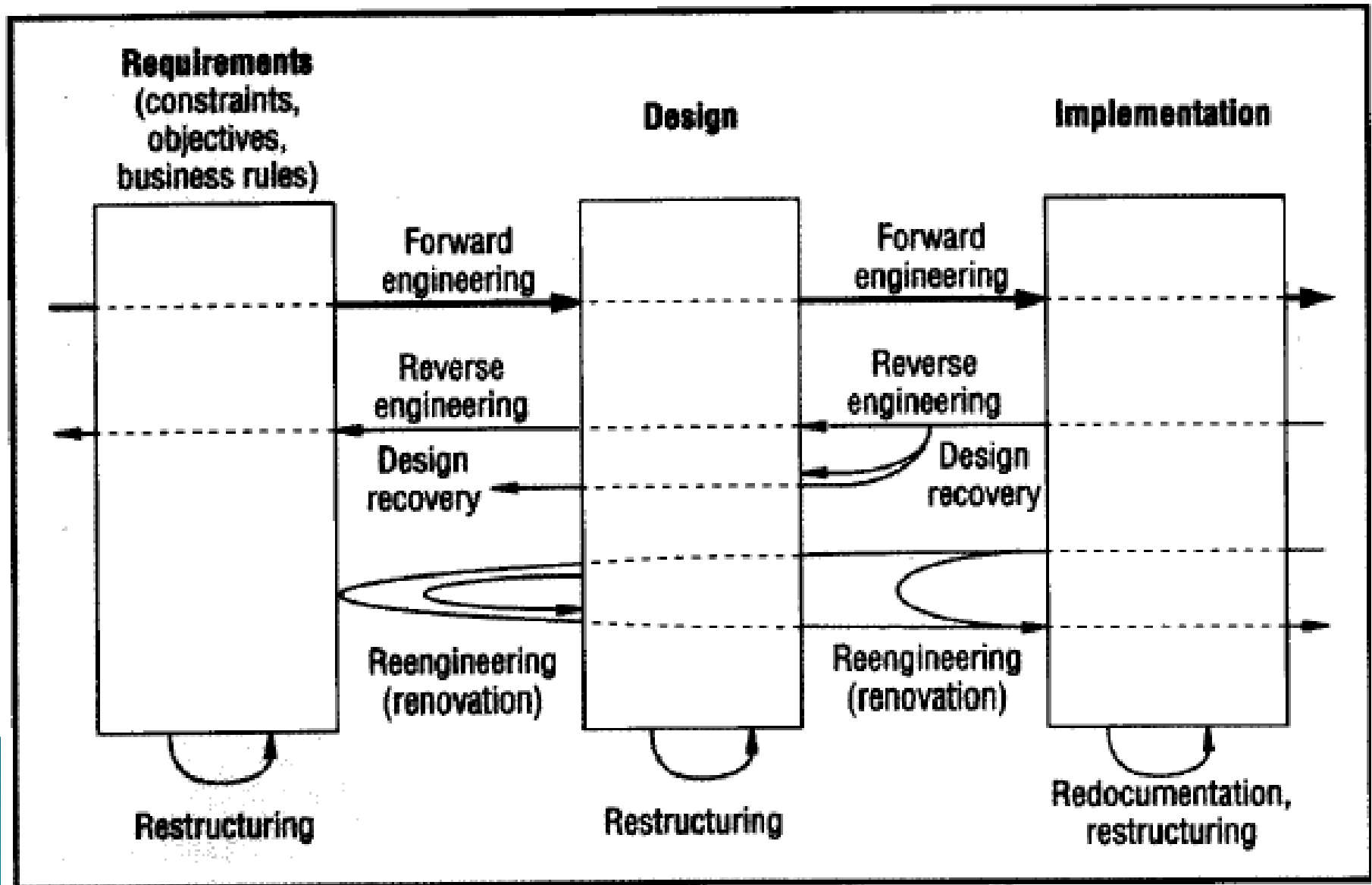


1

0..*



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

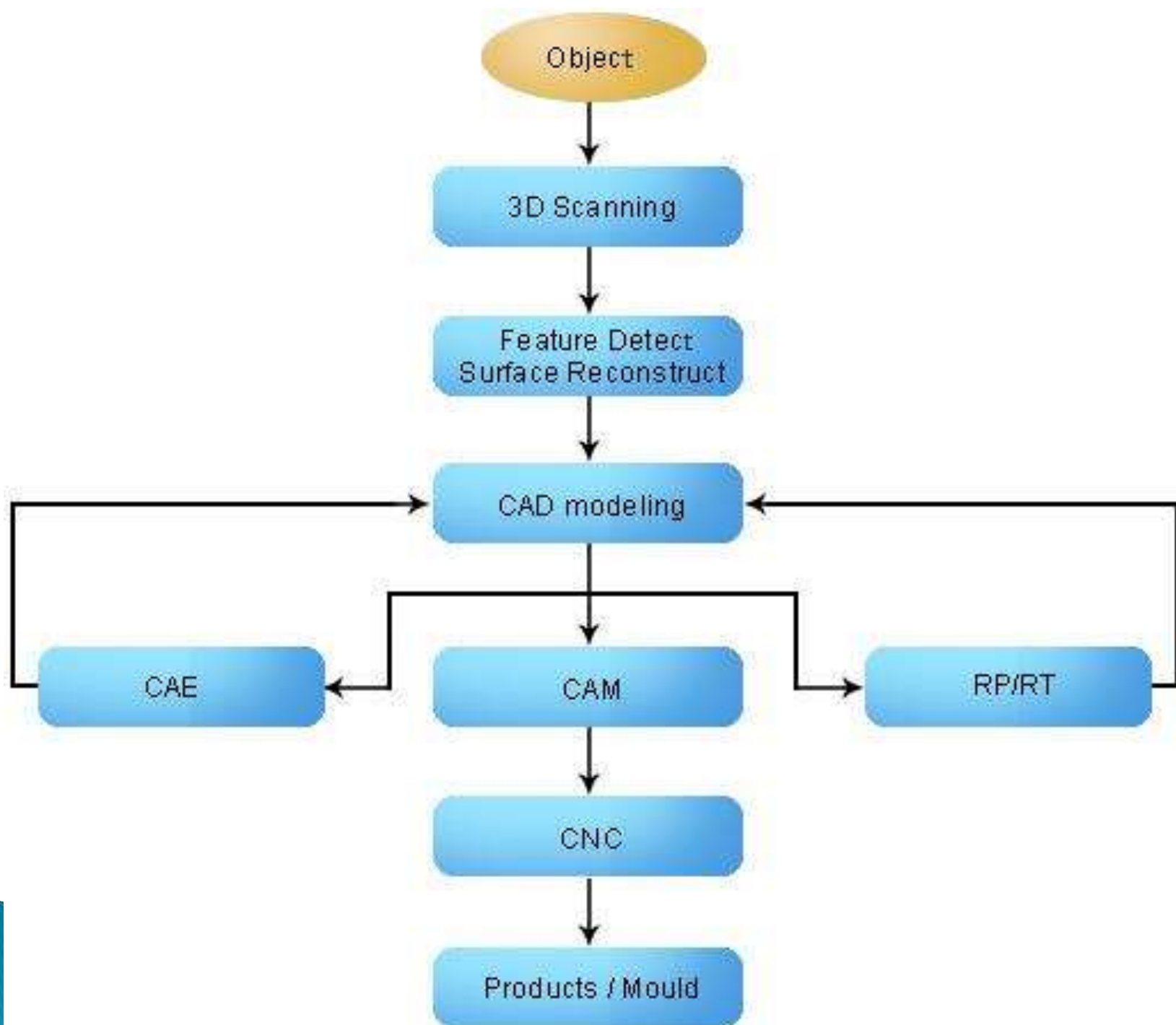
- ▶ Interoperability
- ▶ Lost documentation
- ▶ Product analysis
- ▶ Security auditing
- ▶ Removal of copy protection, circumvention of access restrictions
- ▶ Creation of unlicensed/unapproved duplicates
- ▶ Academic/learning purposes
- ▶ Curiosity
- ▶ Competitive technical intelligence (understand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

Types of RE

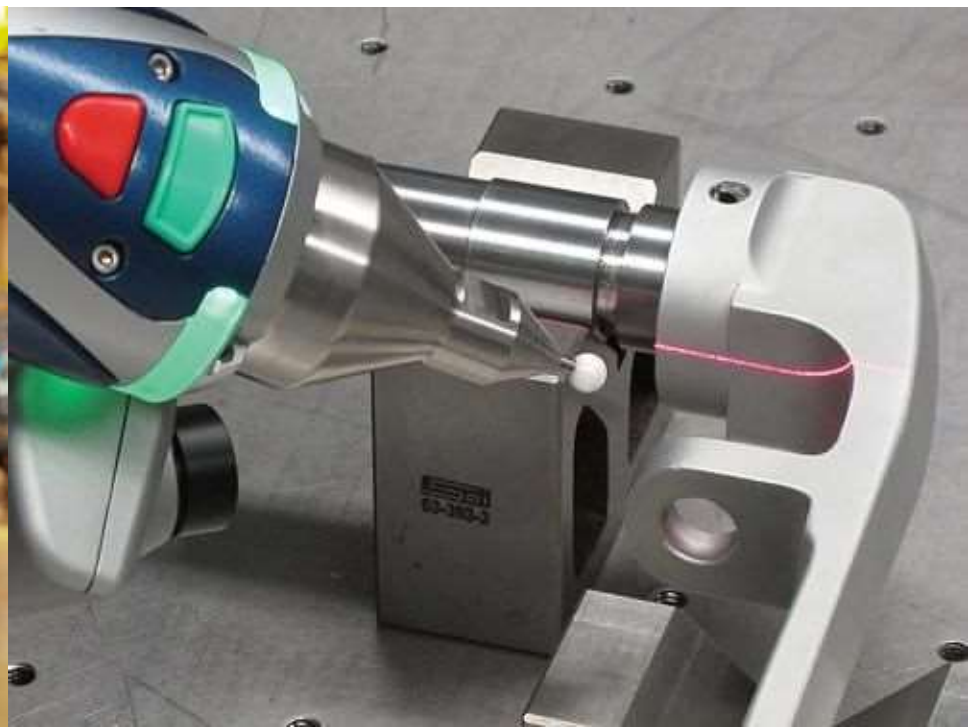
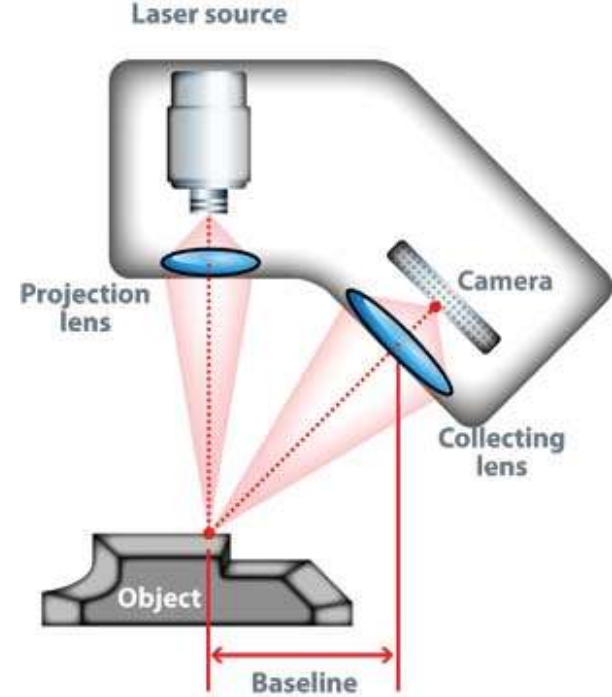
- ▶ Reverse engineering of mechanical devices
- ▶ Reverse engineering of integrated circuits/smart cards
- ▶ Reverse engineering for military applications
- ▶ Reverse engineering of software

Reverse engineering of mechanical devices

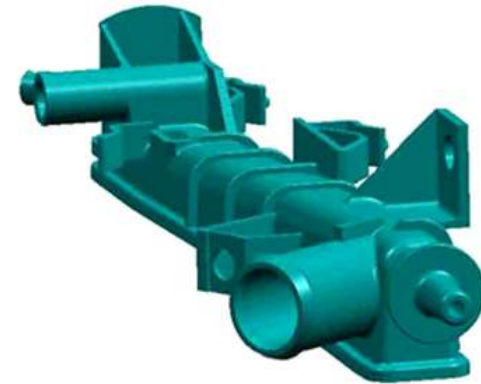
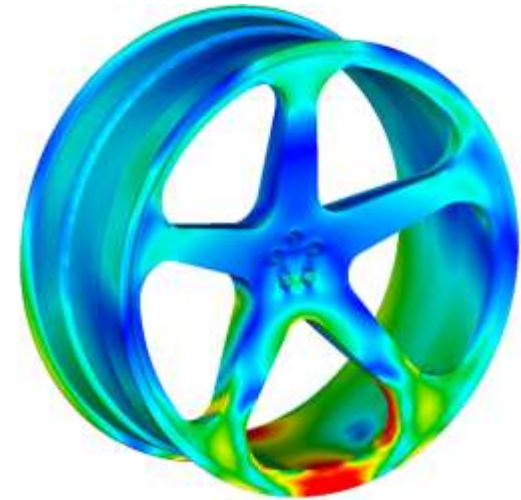
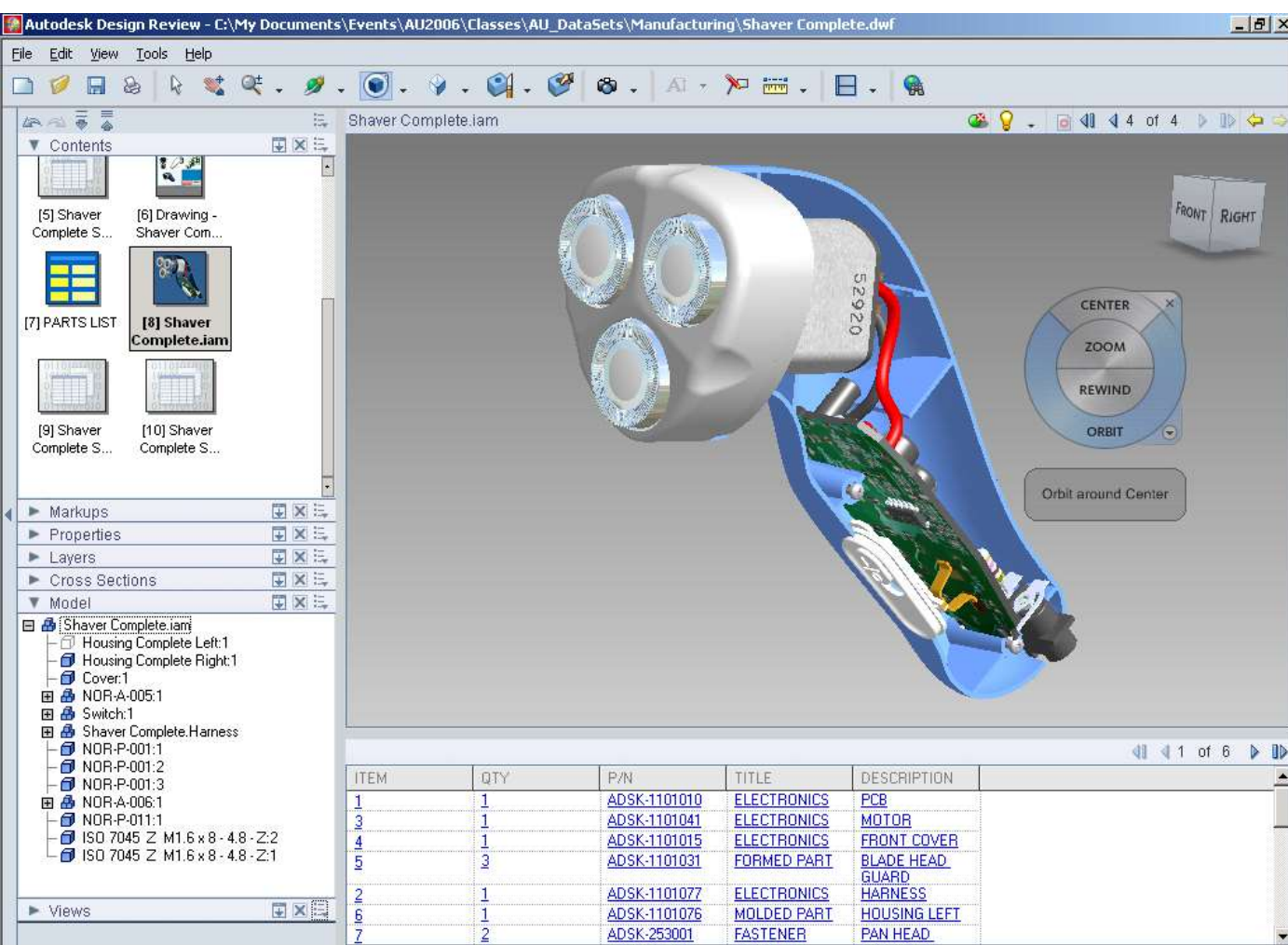
- ▶ Involves measuring an object and then reconstructing it as a 3D model
- ▶ The physical object can be measured using 3D scanning technologies like CMMs, laser scanners, structured light digitizers or computed tomography



Scanere laser 3D



Servicii de modelare 3D CAD



Servicii de imprimare 3D


- ▶ Rapid prototyping

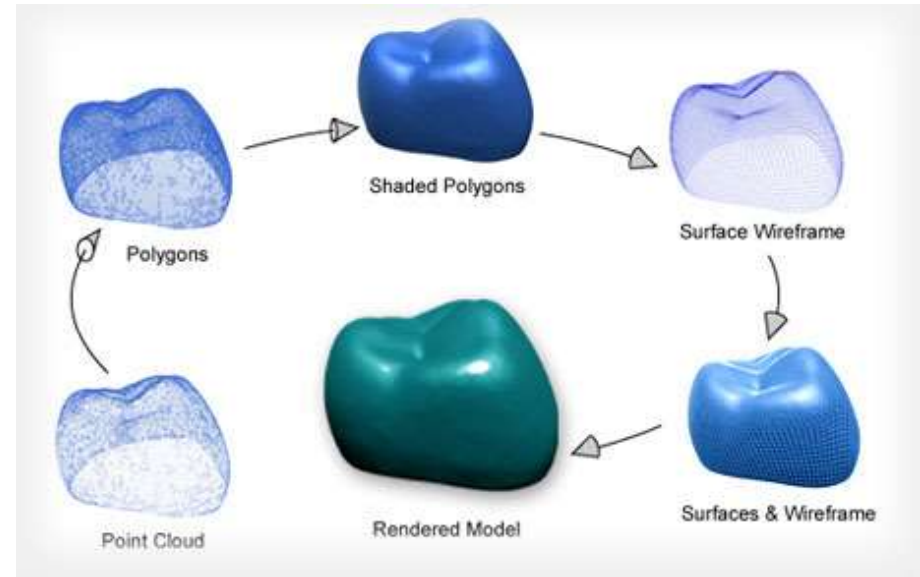


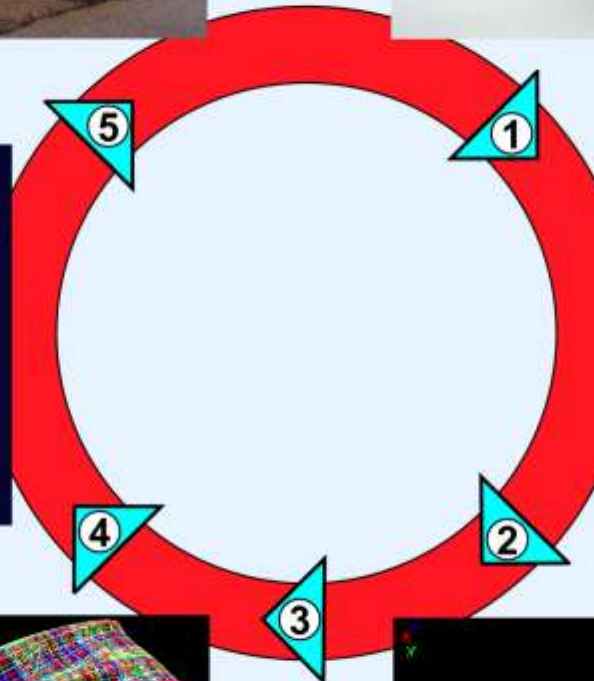
- ▶ FullCure materials



Domenii

Physical Part	Modeling Data
	
	
	



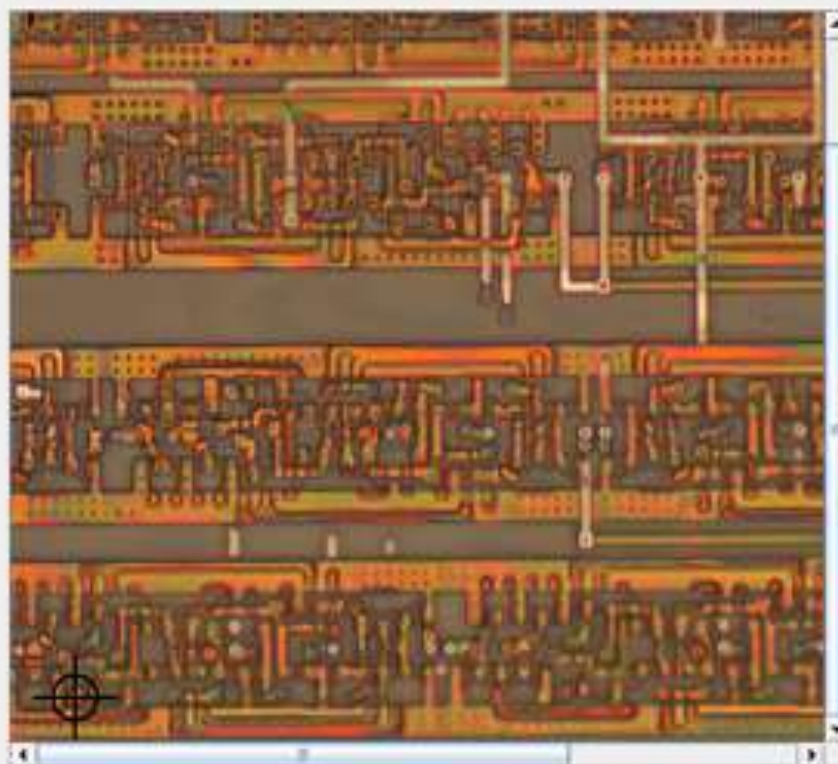
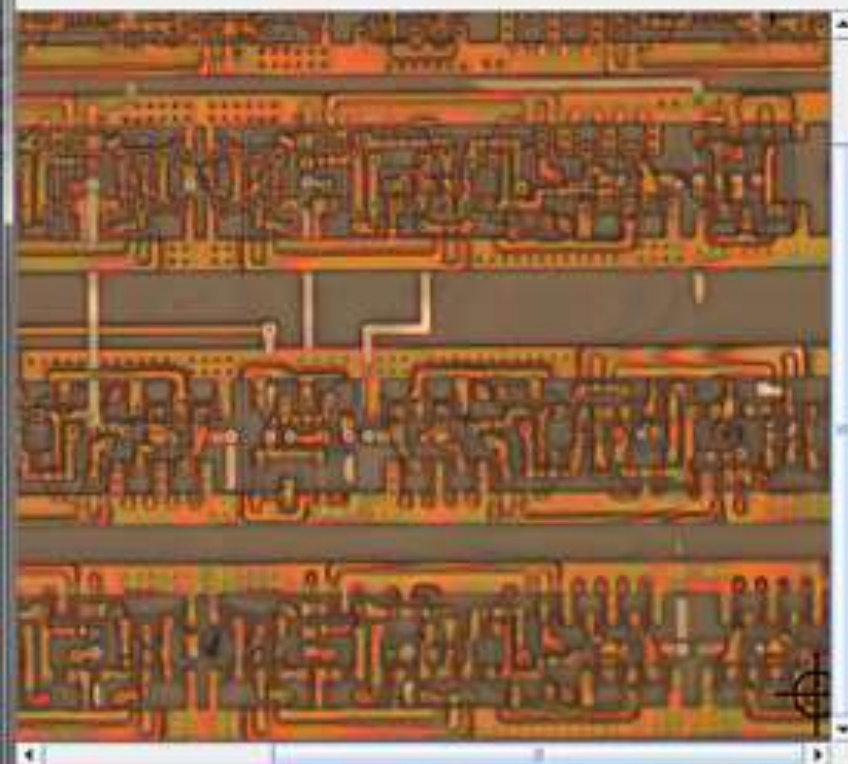


Reverse engineering of integrated circuits/smart cards

- ▶ 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... \...Praktika\CREla\ChipBilder\iz...



654 - 63

580 - 484



12 - 64

38 - 486



The screenshot shows the degate software interface. The background is a grayscale circuit layout with various logic gates and interconnects. A ruler at the top indicates coordinates from 5300 to 6600. Two dialog boxes are open in the foreground.

Logic gates dialog:

Short Name	#	Width	Height	Fill color	Frame color	Description
01-FF	58	272	134			D-Q-FlipFlop
02-FF	11	343	134			D-Q-FlipFlop with rst
03-FF	17	343	133			D-Q-FlipFlop with rts
04-BUF	5	226	128			
05-3XOR	13	249	133			
06: 2-XNOR	12	133	133			

Buttons: Edit, Add, Remove, Close

Edit gate dialog:

Entity Behaviour Layout

Short name: 02-FF

Description: D-Q-FlipFlop with rst

Logic Class: flipflop (generic)

Port ID	Port Name	Port Description	In	Out
2384	!Q	!Q	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2380	Q	Q	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2388	clk	clk	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2386	D	D	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2382	rst	rst	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Buttons: Add, Remove

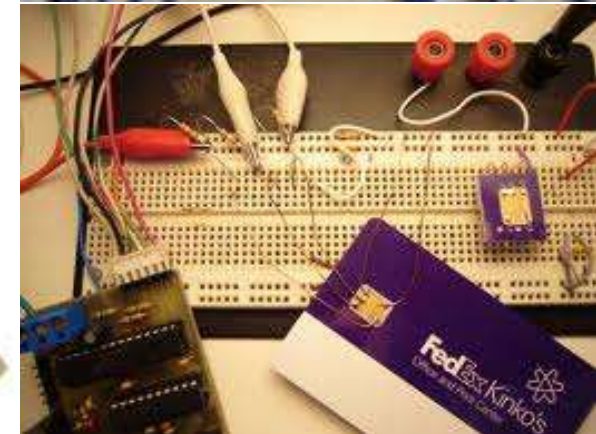
Fill color: [Color swatch] Reset Color

Frame color: [Color swatch] Reset Color

Buttons: Cancel, OK

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

Avioane

► US – B-29

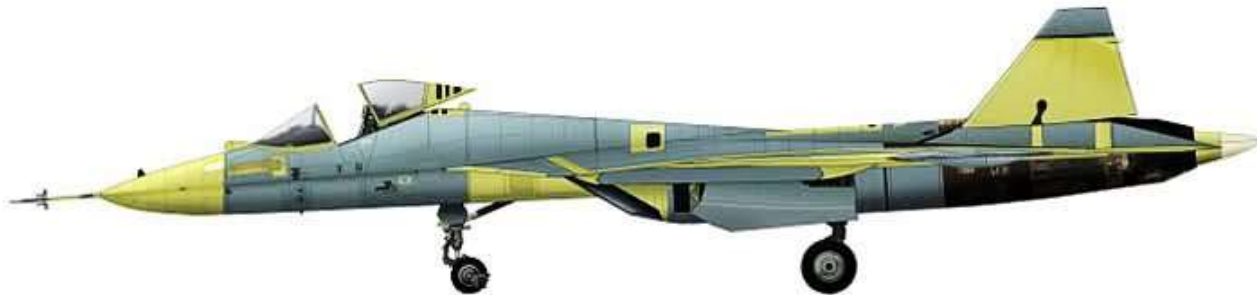
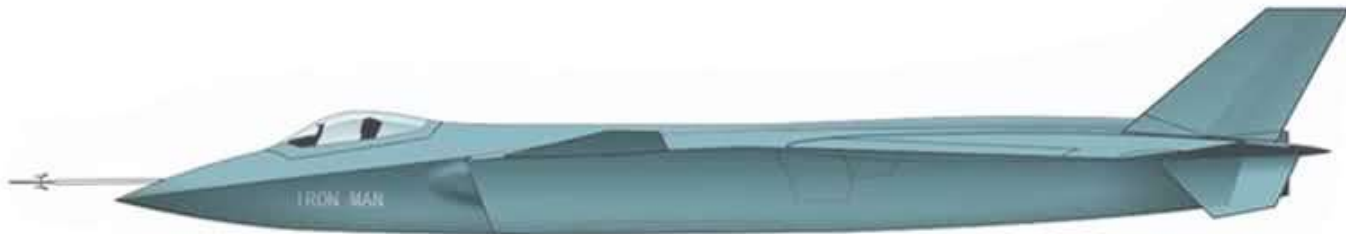


URSS – Tupolev Tu-4



Avioane (2)

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



Rachete

- ▶ US – AIM-9 Sidewinder
- Soviet – Vympel K-13



Submarine

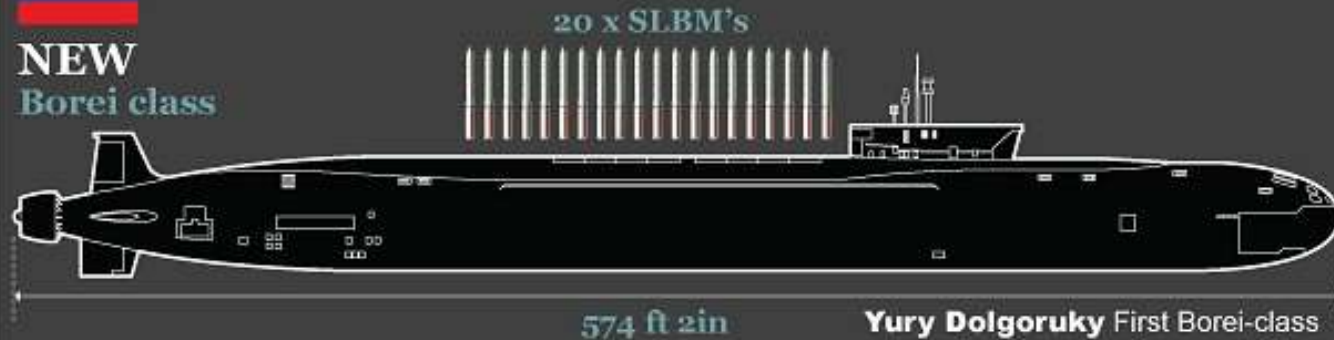
Russia's new ballistic missile submarine



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



UFOs

United States Patent (19)
(Prior)

1911 3,774,865
1912 Nov. 27, 1913

[T4] **Author:** Olegário F. Flatto, Rua Vinte e de Quatro, 63, Rio De Janeiro.

1210 Filed Jan. 3, 1972
1211 Appl. No. 316,618

Related U.S. Applications

[50] Classification of Gr. No. 4534, 1922-23.

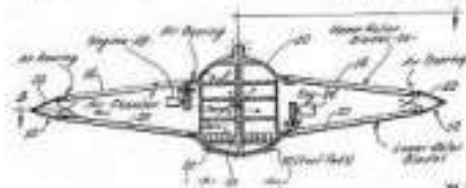
25. 1999

Primary Physician—Diane A. Siegel
Assistant Physician—Joan D. Smith
Secretary—Kath D. Newkirk

2002

ABSTRACT

WALLACE
FUR GENERATING A SECONDARY
NAIL FORCE FIELD



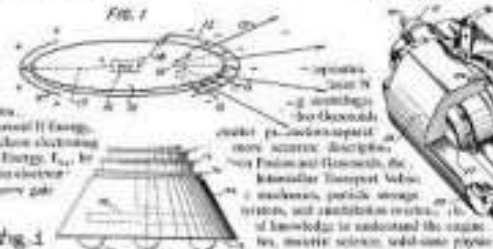
Feb 20, 1962

T. T. BRIDGE
 EXECUTIVE DIRECTOR

3.0022.4

Received July 8, 2003

B. Baccus—Plum. 3



May 2001, 13001

ESCU311

INTERSTELLARITES

Reverse Engineering a UFO



by Robert Klein

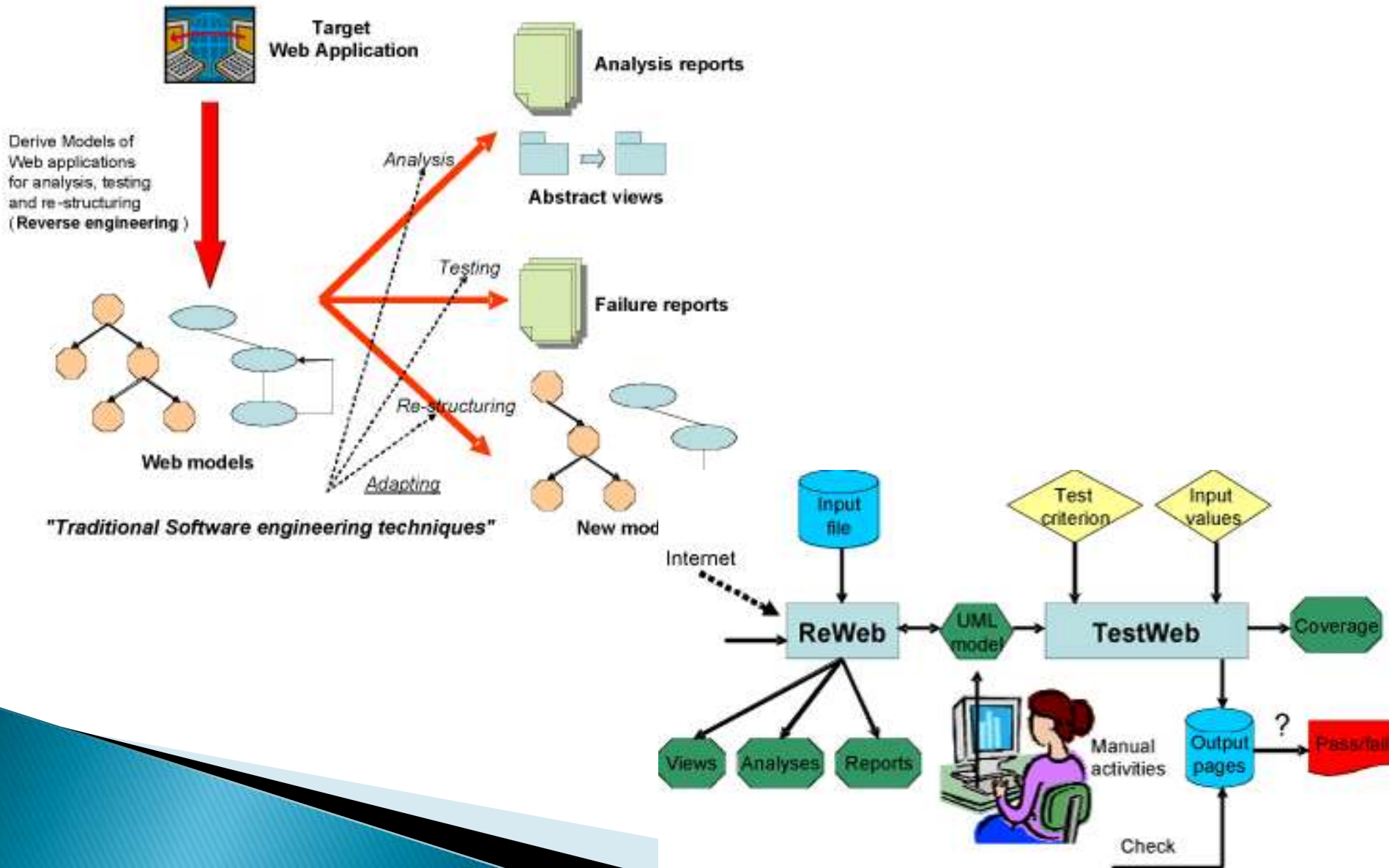
Smart phones



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

RE of Web Applications



Other purposes of RE for software

- ▶ security auditing,
- ▶ removal of copy protection ("cracking"),
- ▶ circumvention of access restrictions often present in consumer electronics,
- ▶ customization of embedded systems (such as engine management systems),
- ▶ in-house repairs or retrofits,
- ▶ enabling of additional features on low-cost "crippled" hardware (such as some graphics card chipsets),
- ▶ or even mere satisfaction of curiosity.

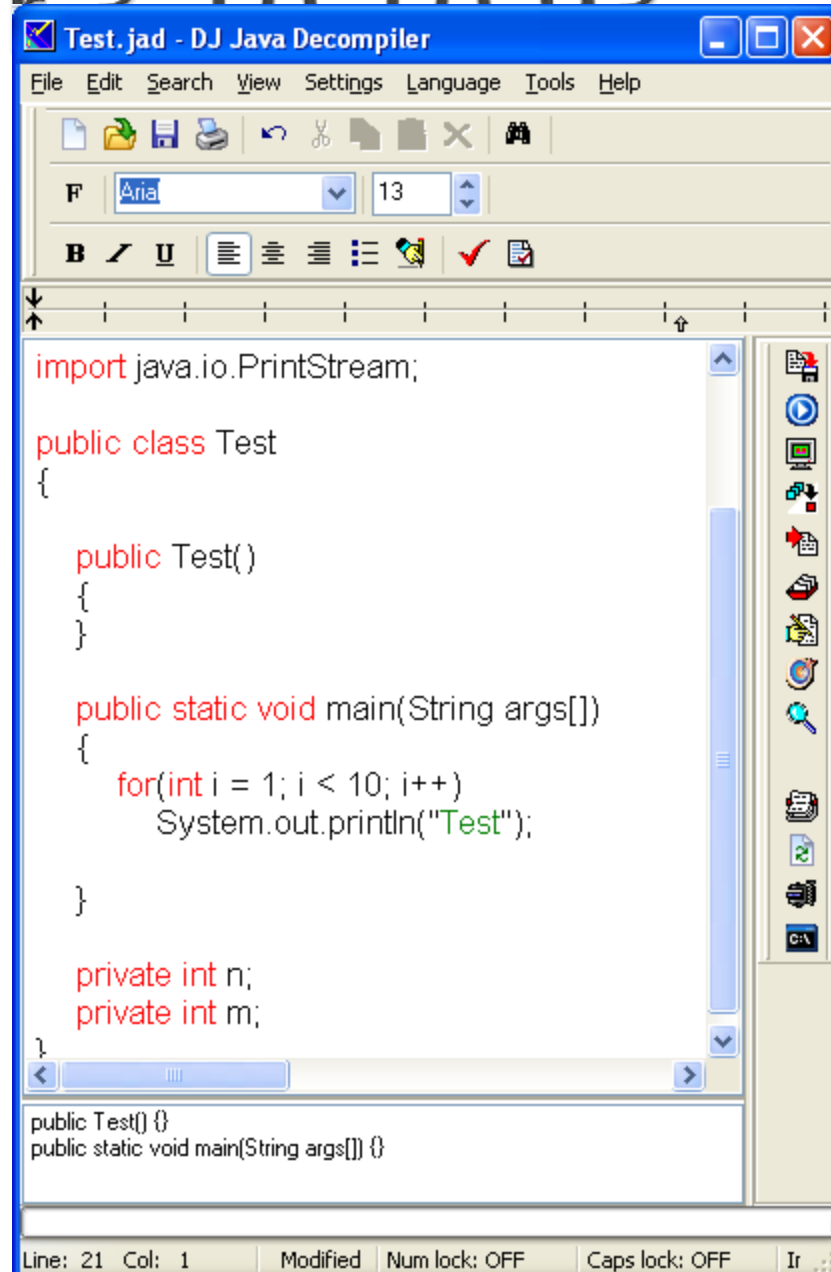
Binary Software – Reverse Code Engineering

- ▶ Decompilation of binaries for the Java platform can be accomplished using Jad or DJ Decompiler
- ▶ The Samba software, which allows systems that are not running Microsoft Windows systems to share files with systems that are
- ▶ OpenOffice.org is one party doing this for the Microsoft Office file formats

DJ Java Decompiler 2.10.10.02

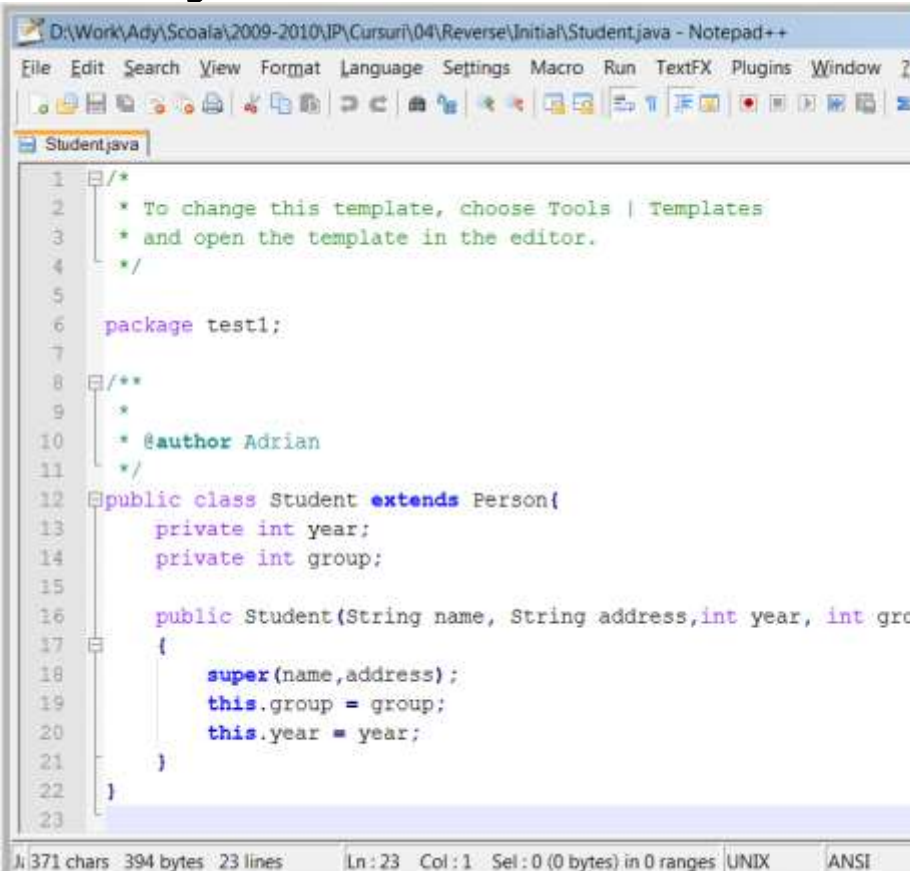
```
public class Test
{
    private int n;
    private int m;

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



JAD

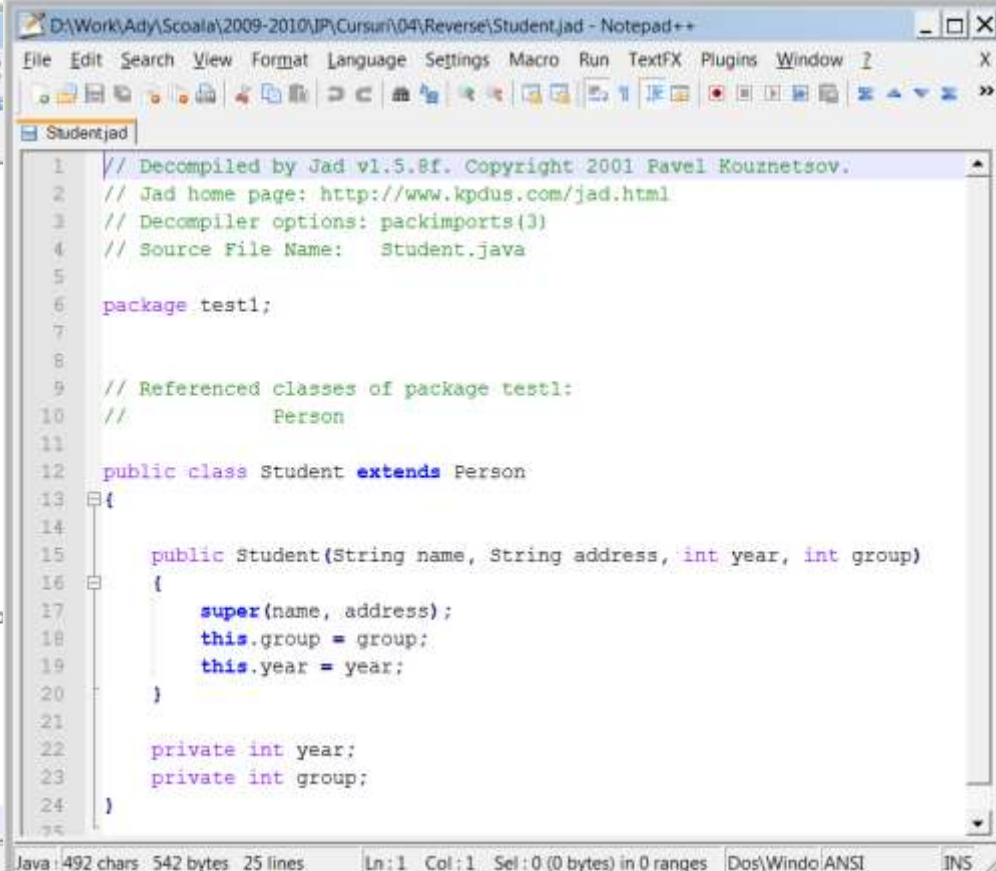
- ▶ Link: <http://www.steike.com/code/java-reverse-engineering/>
- ▶ `jad.exe NumeFisier.class => NumeFisier.jad`



The screenshot shows a Notepad++ window titled "D:\Work\Ady\Scoala\2009-2010\IP\Cursuri\04\Reverse\Initial\Student.java - Notepad++". The file "Student.java" is open, displaying the following code:

```
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
```

The status bar at the bottom indicates: 371 chars 394 bytes 23 lines Ln: 23 Col: 1 Sel: 0 (0 bytes) in 0 ranges UNIX ANSI.



The screenshot shows a Notepad++ window titled "D:\Work\Ady\Scoala\2009-2010\IP\Cursuri\04\Reverse\Student.jad - Notepad++". The file "Student.jad" is open, displaying the following decompiled code:

```
1  // Decompiled by Jad vl.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
9  // Referenced classes of package test1:
10 //     Person
11
12  public class Student extends Person
13  {
14
15      public Student(String name, String address, int year, int group)
16      {
17          super(name, address);
18          this.group = group;
19          this.year = year;
20      }
21
22      private int year;
23      private int group;
24  }
```

The status bar at the bottom indicates: Java 492 chars 542 bytes 25 lines Ln: 1 Col: 1 Sel: 0 (0 bytes) in 0 ranges Dos\Windo ANSI JNS.

Open Office

The screenshot displays the OpenOffice.org Impress application window. The title bar reads "ss - OpenOffice.org Impress". The menu bar includes "File", "Edit", "View", "Insert", "Format", "Tools", "Slide Show", "Window", and "Help". The toolbar contains various icons for file operations, editing, and presentation controls. The "Slides" panel on the left shows a thumbnail of the current slide, labeled "Slide 1". The main slide area has a light blue background and contains the following elements:

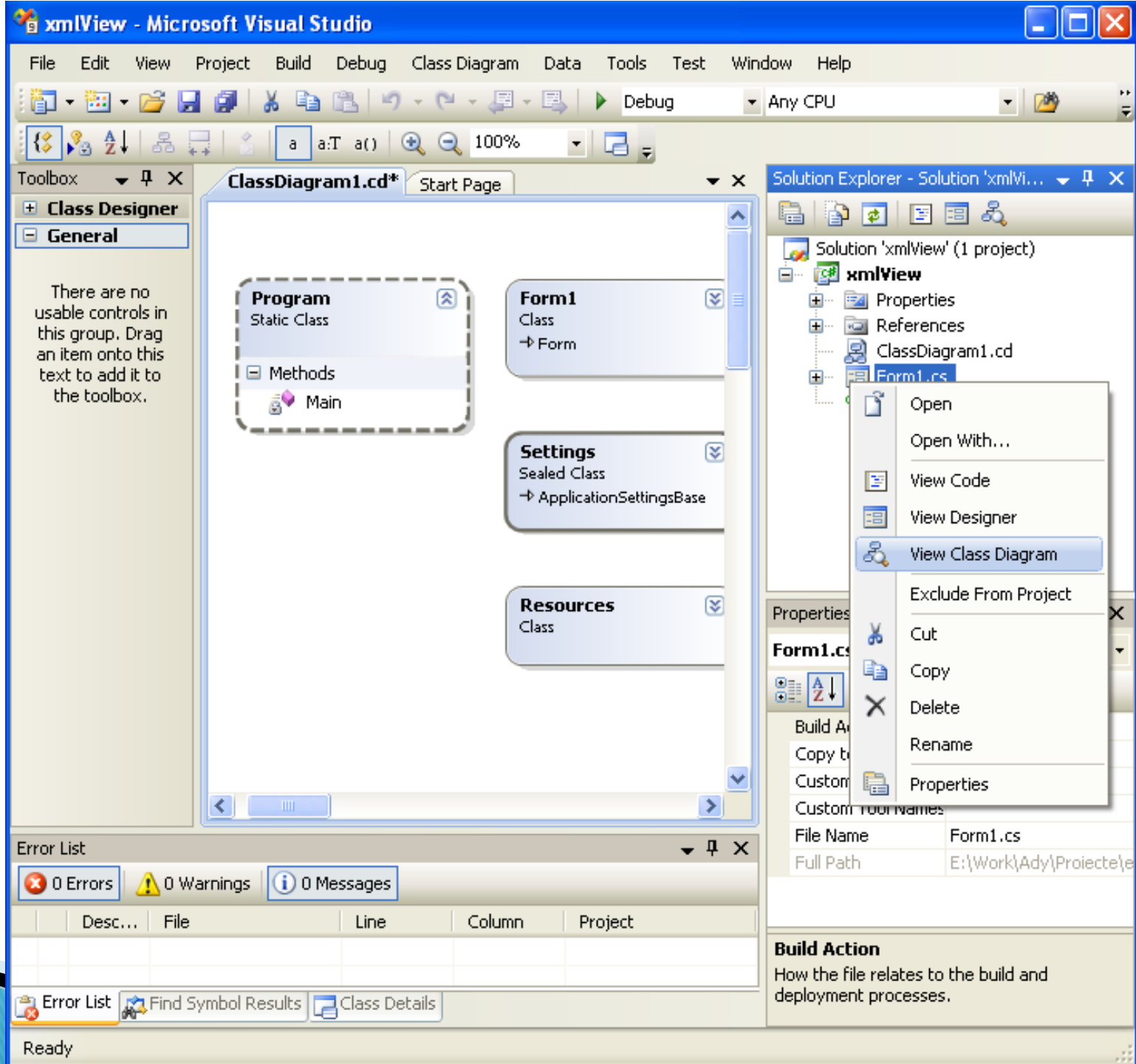
- A blue header bar with the text "Exemplu OpenOffice.org".
- A bulleted list on the left side:
 - Alt exemplu
 - Și altul
 - Și încă unul
 - Și încă unul
- A pie chart on the right side, divided into five segments of different colors: dark blue, green, yellow, red, and light blue.
- Three yellow stars of varying sizes at the bottom left of the slide.

The "Tasks" panel on the right shows a "Layouts" section with various slide templates. The status bar at the bottom indicates "Page 1 / 1", "29,16 / -2,59", "0,00 x 0,00", "51%", "Slide 1 / 1", and "Default".

Binary software techniques

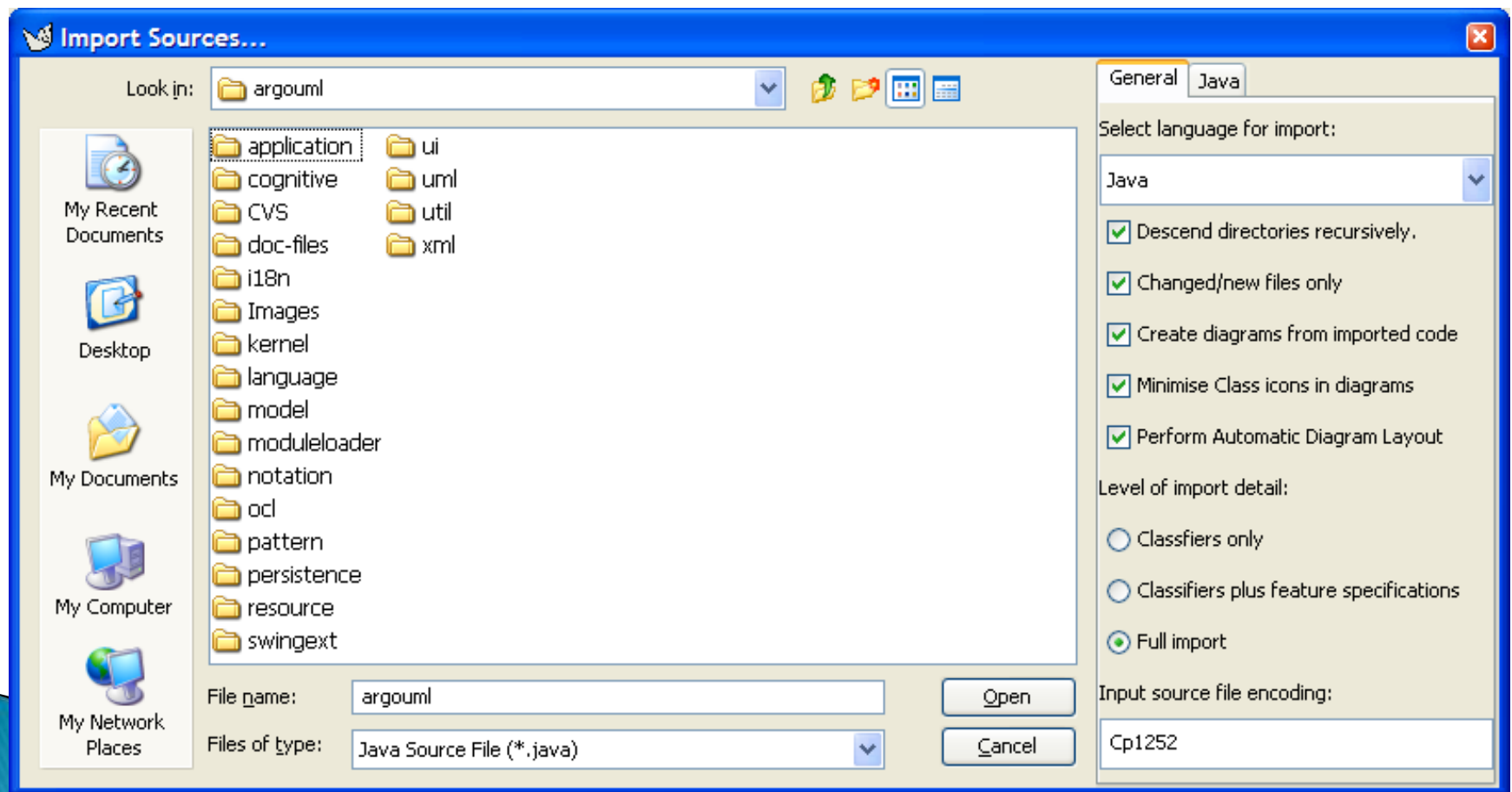
- ▶ Analysis through observation of information exchange (bus analyzers and packet sniffers, for example, for accessing a computer bus or computer network connection)
- ▶ Disassembly using a disassembler
- ▶ Decompilation using a decompiler (try to recreate the source code in some high-level language for a program only available in machine code or bytecode)

C#

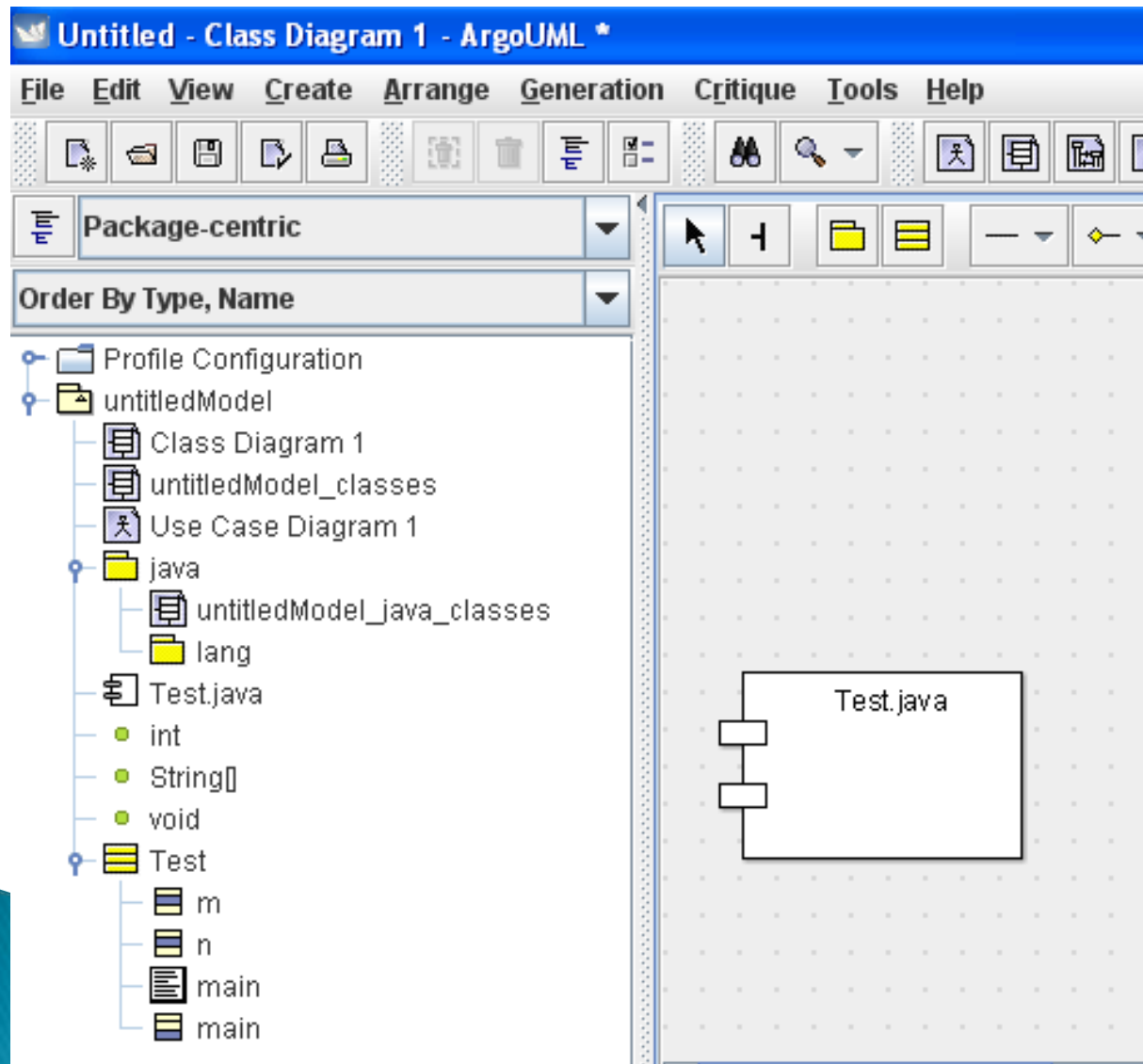


RE în ArgoUML

► File -> Import Sources...



Pentru exemplul anterior...



Demo 1

- ▶ **Forward engineering:**
 - Diagrame de clasă -> .java files (ArgoUML)
 - .java files -> .class files (NetBeans)
- ▶ **Reverse engineering:**
 - .class files -> .java files (JAD Decompiler)
 - .java files -> Diagrame de Clasă (ArgoUML)

UML2.0 – 13 Tipuri de Diagrame

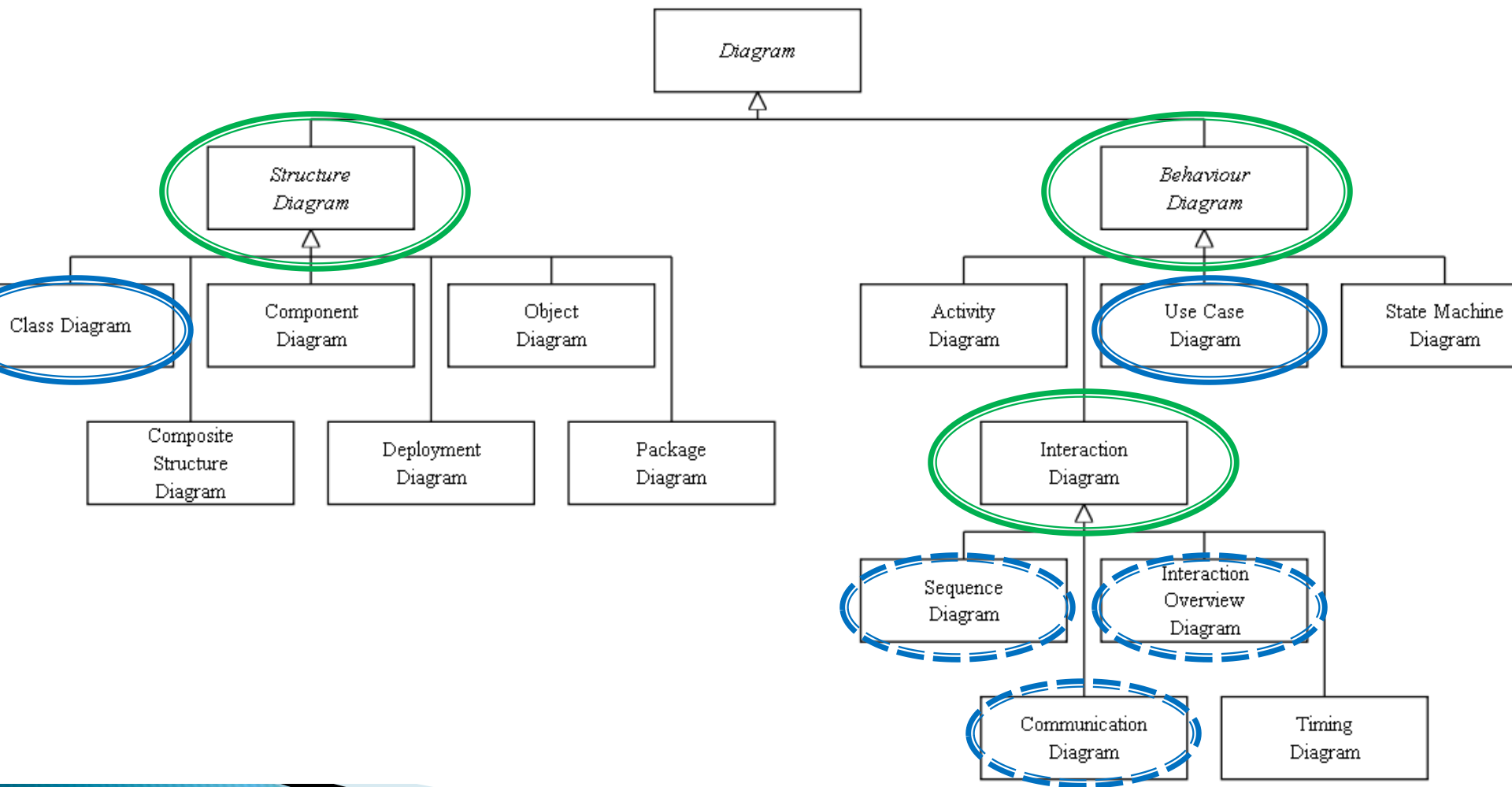


Diagrama de Interacțiuni 1

- ▶ Ilustrează cum **interacționează** (colaborează, comunică) obiectele între ele cu ajutorul mesajelor
- ▶ Folosită pentru a modela **comportamentul** unei mulțimi de obiecte dintr-un anumit context care interacționează în vederea îndeplinirii unui anumit scop
- ▶ **Scop:** specifică modul în care se realizează o operație sau un caz de utilizare

Diagrama de Interacțiuni 2

- ▶ Contextul unei interacțiuni:
 - Sistem (subsistem)
 - Operație
 - Clasă
- ▶ Obiectele:
 - Pot fi lucruri concrete sau prototipuri între ele
 - Se pot stabili conexiuni semantice (legături)
 - Comunică între ele prin schimburi de mesaje

Mesaj

- ▶ Specifică o comunicare între obiecte
- ▶ Îi este asociată o acțiune care poate avea ca efect schimbarea stării actuale a obiectului
- ▶ Forma generală a unui mesaj:
[cond garda] acțiune (lista parametrilor)

Tipuri de acțiuni în UML

- ▶ **call**: invocă o operație a unui obiect
- ▶ **return**: returnează o valoare apelantului
- ▶ **send**: trimite un semnal
- ▶ **create**: creează un obiect
- ▶ **destroy**: distruge un obiect

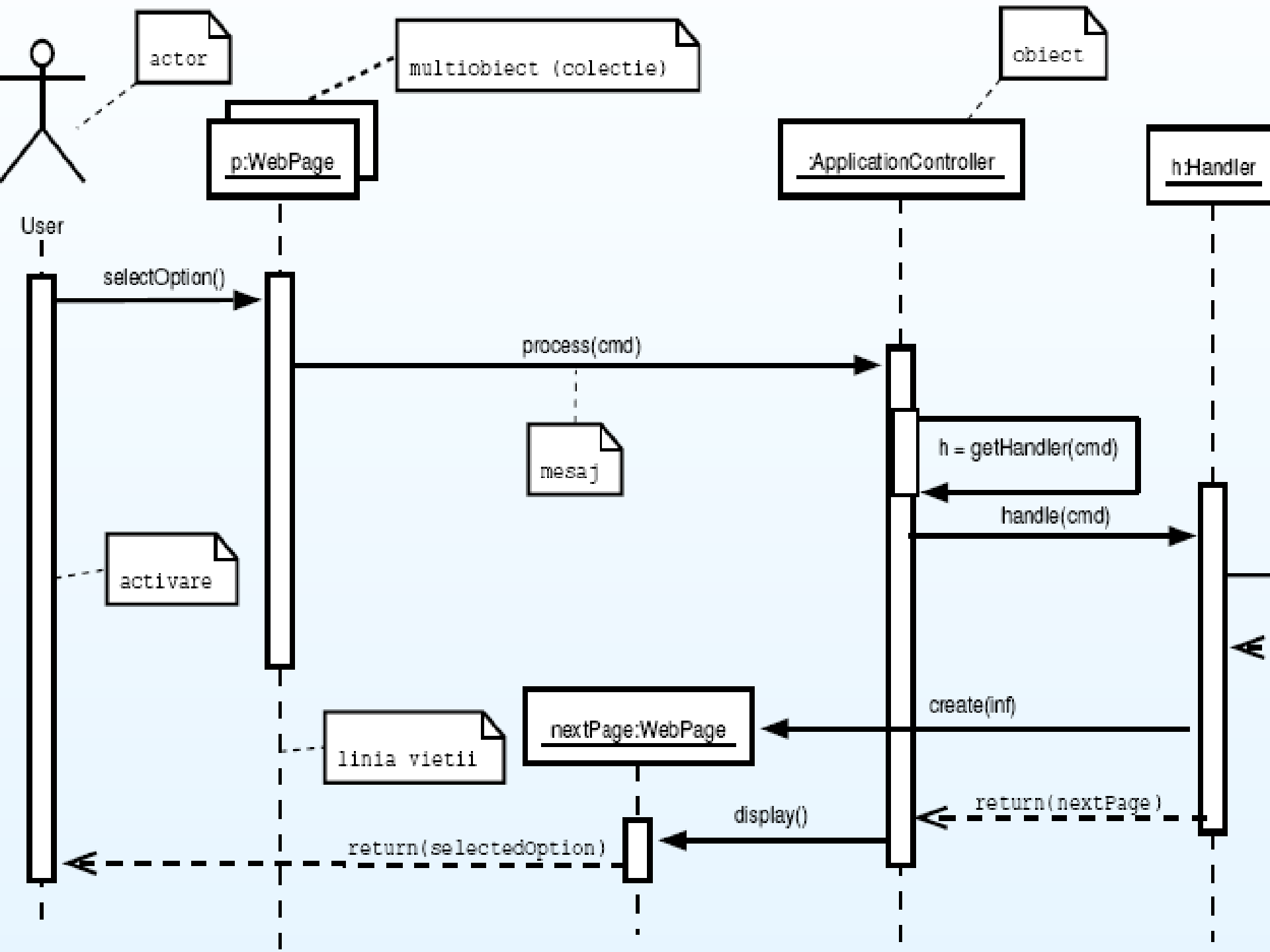
Diagrama de Interacțiuni 3

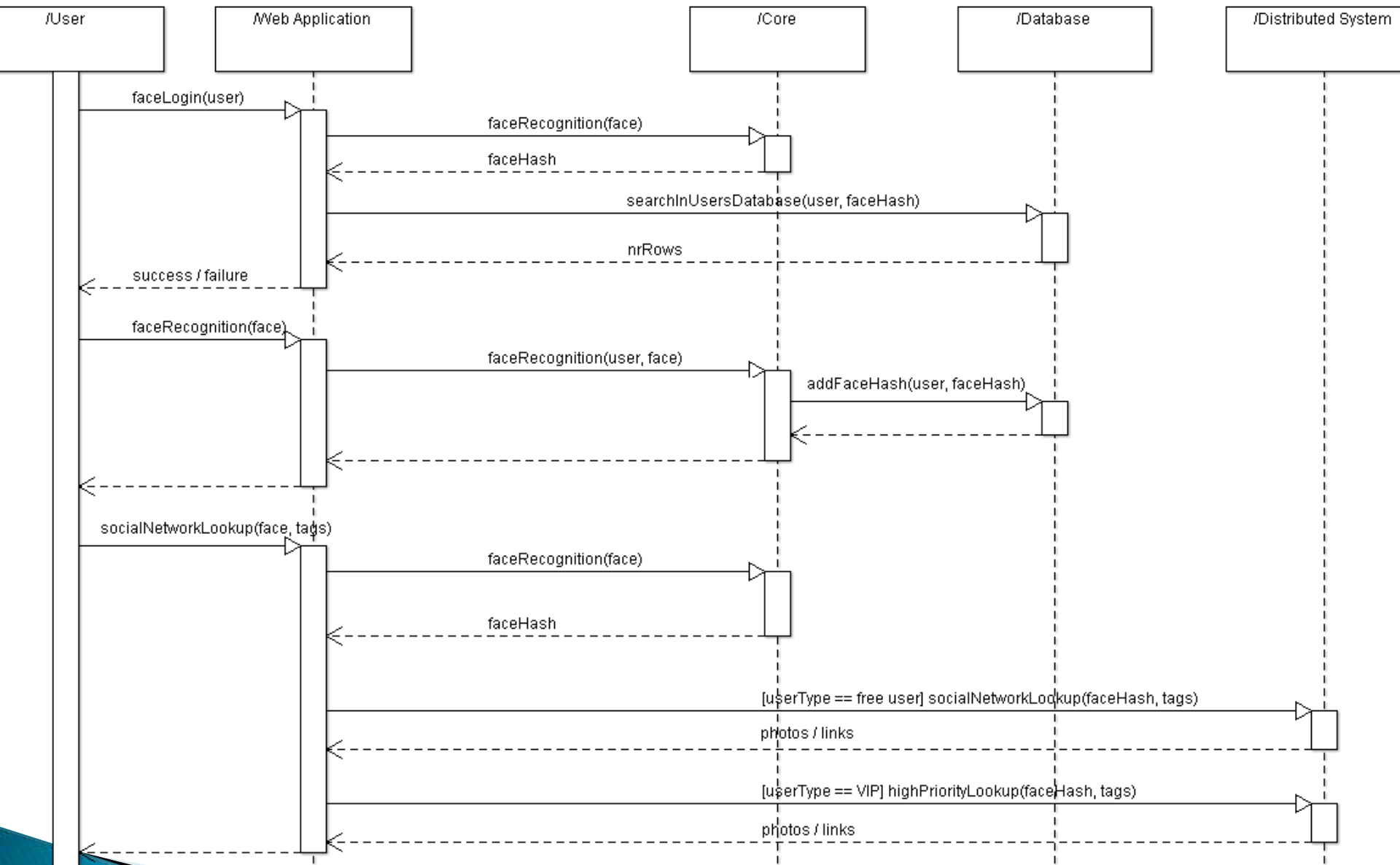
- ▶ Poate conține:
 - Obiecte, actori, clase
 - Relații
 - Mesaje
- ▶ Tipuri de diagrame de interacțiuni:
 - Diagrama de Secvență
 - Diagrama de Colaborare

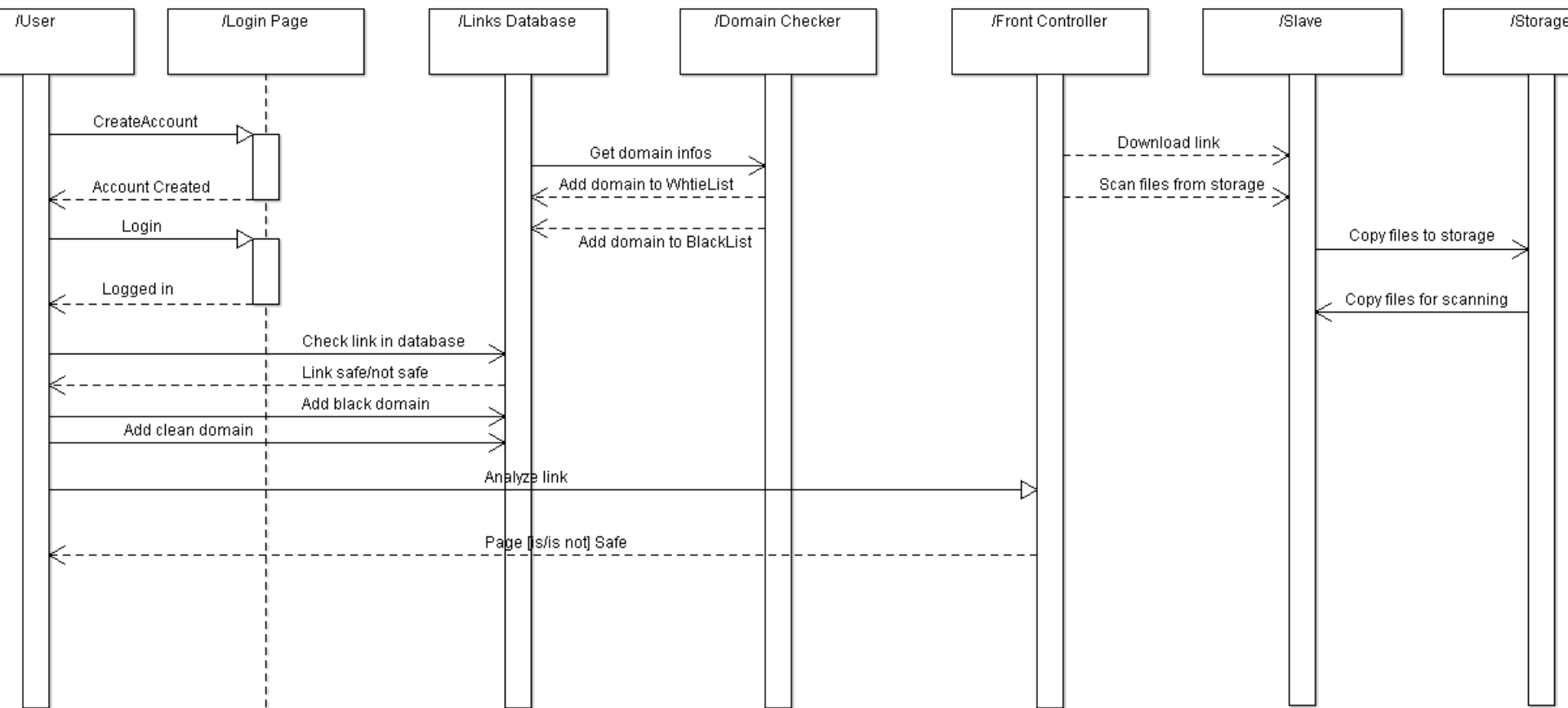
specifică aceeași informație dar pun accentul pe aspecte diferite

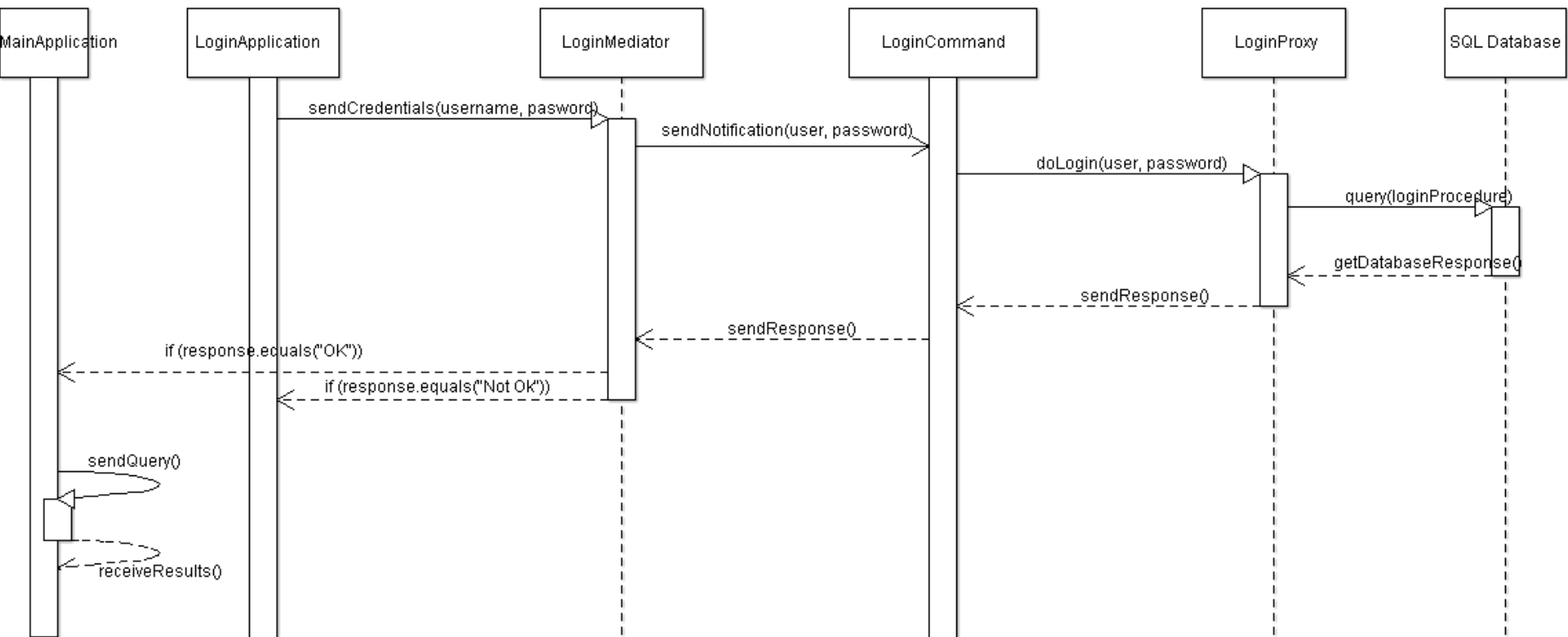
Diagrama de Secvență

- ▶ **Diagrama de secvență** curprinde 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)










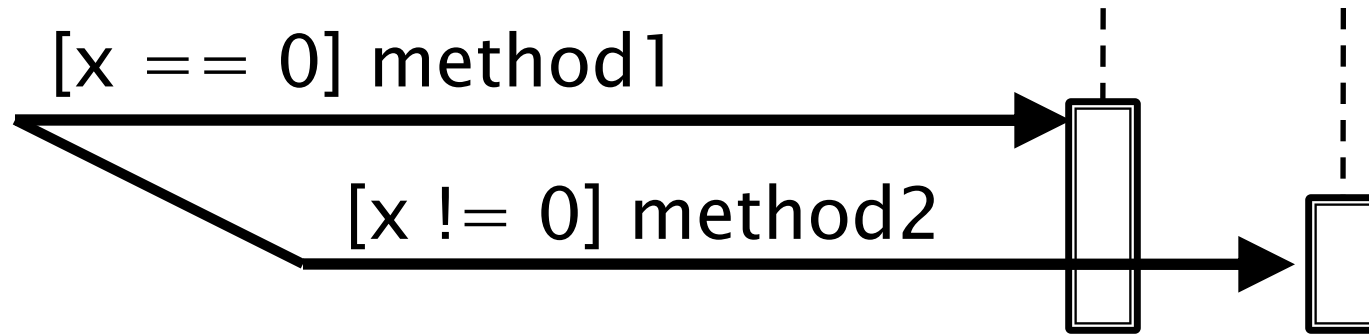


Tipuri de comunicări

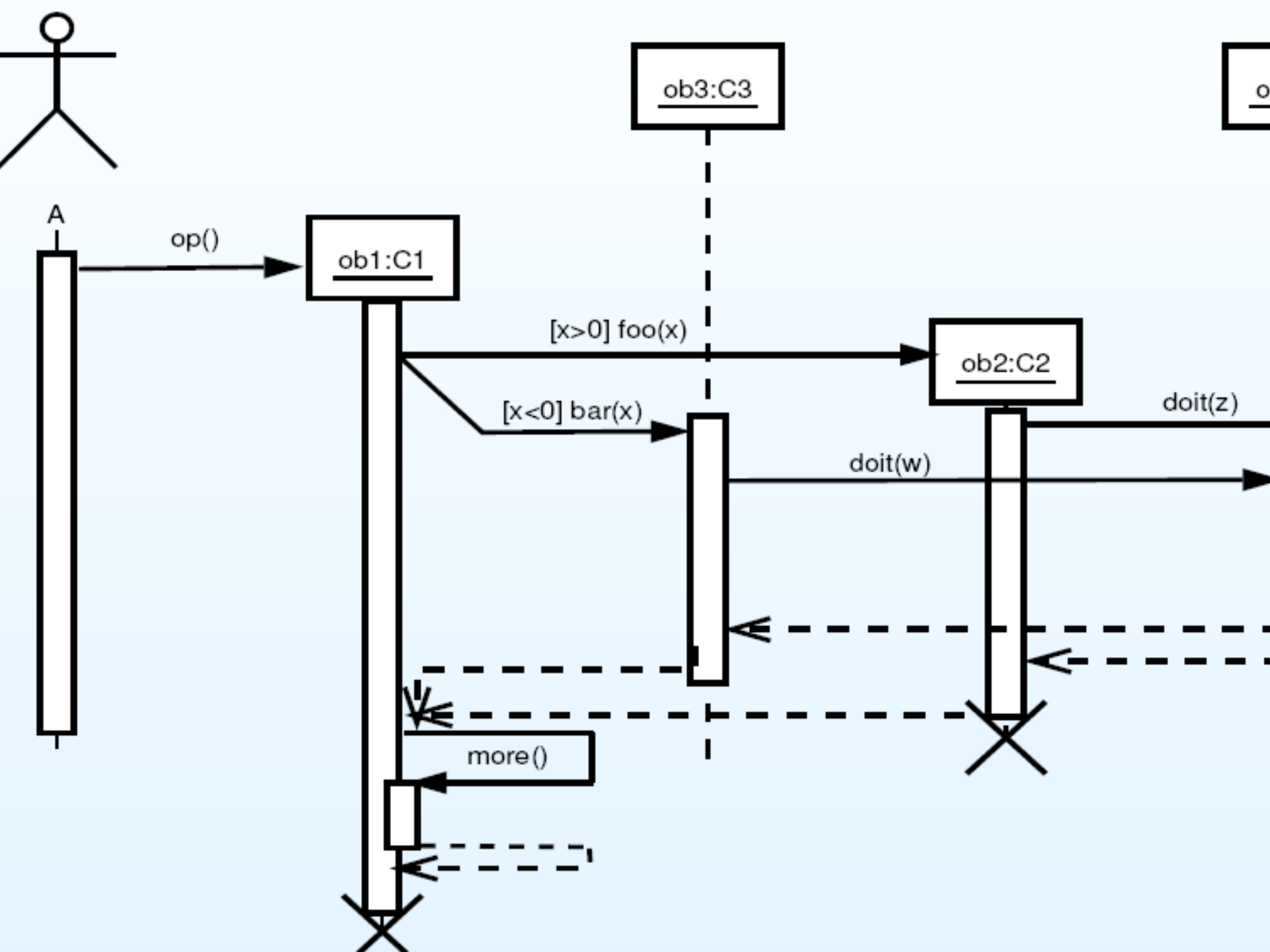
		
Comunicare sincronă	Comunicare asincronă	Return

- ▶ **Sincronă:** controlul execuției trece de la A la B și revine la A după ce B își termină execuția (apel de funcție)
- ▶ **Asincronă:** A trimite un semnal lui B după care își continuă execuția (fire de execuție)
- ▶ **Return:** reîntoarcerea în procedura de unde am plecat

Ramificații

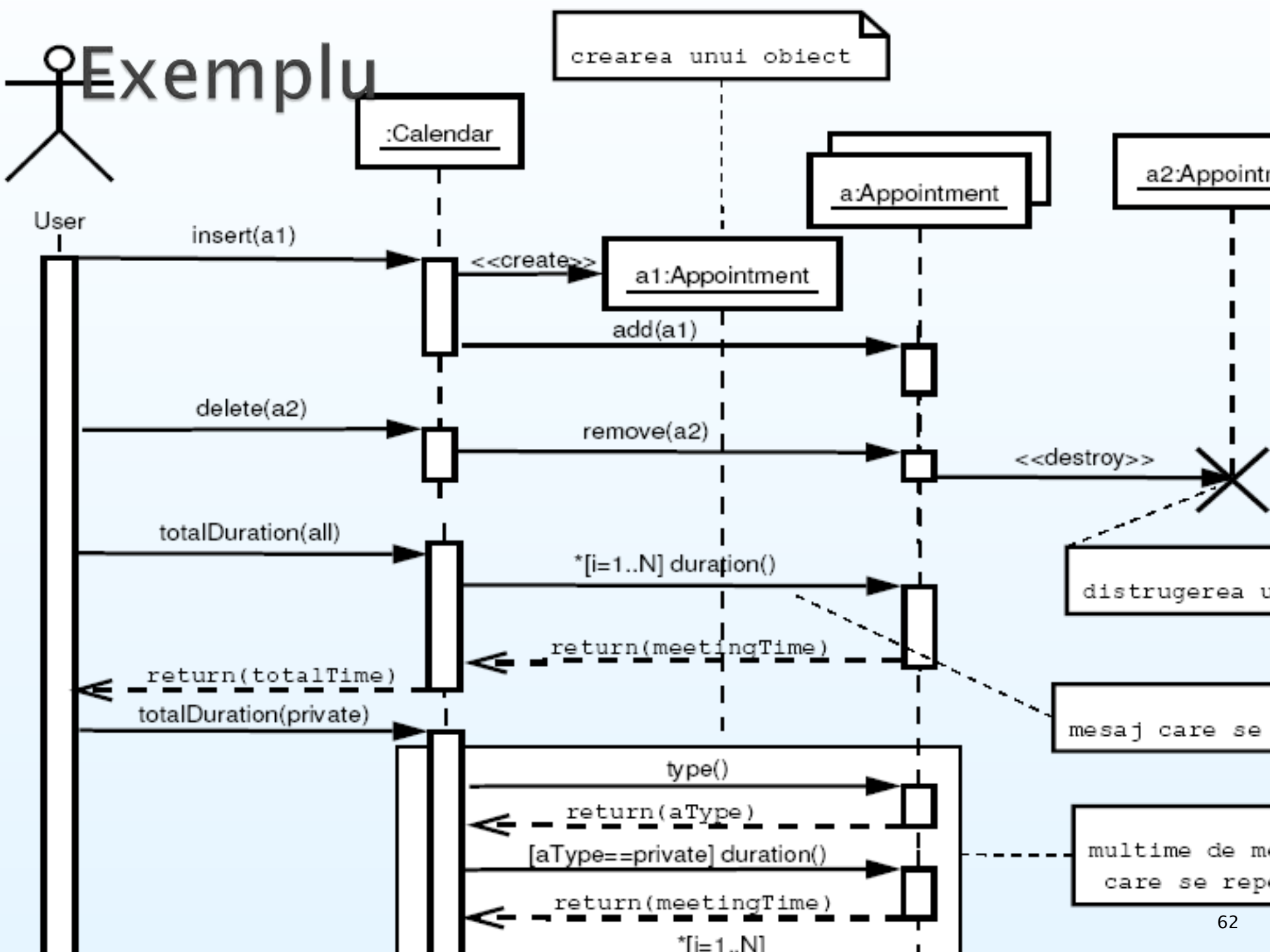


- ▶ Reprezentare: mai multe mesaje care pleacă din același punct și sunt etichetate cu o condiție:
 - condiții mutual exclusive => condiționalitate (if, switch)
 - condiții care se suprapun => concurență



Iterații

- ▶ Indică faptul că un mesaj (o mulțime de mesaje) se repetă
- ▶ Mesajul este etichetat cu o condiție gardă de forma:
- ▶ `*[cond] acțiune(lista parametrilor)`
- ▶ Dacă sunt mai multe mesaje acestea vor fi înconjurate cu un chenar; în interiorul chenarului va fi specificată condiția `(*[cond])`



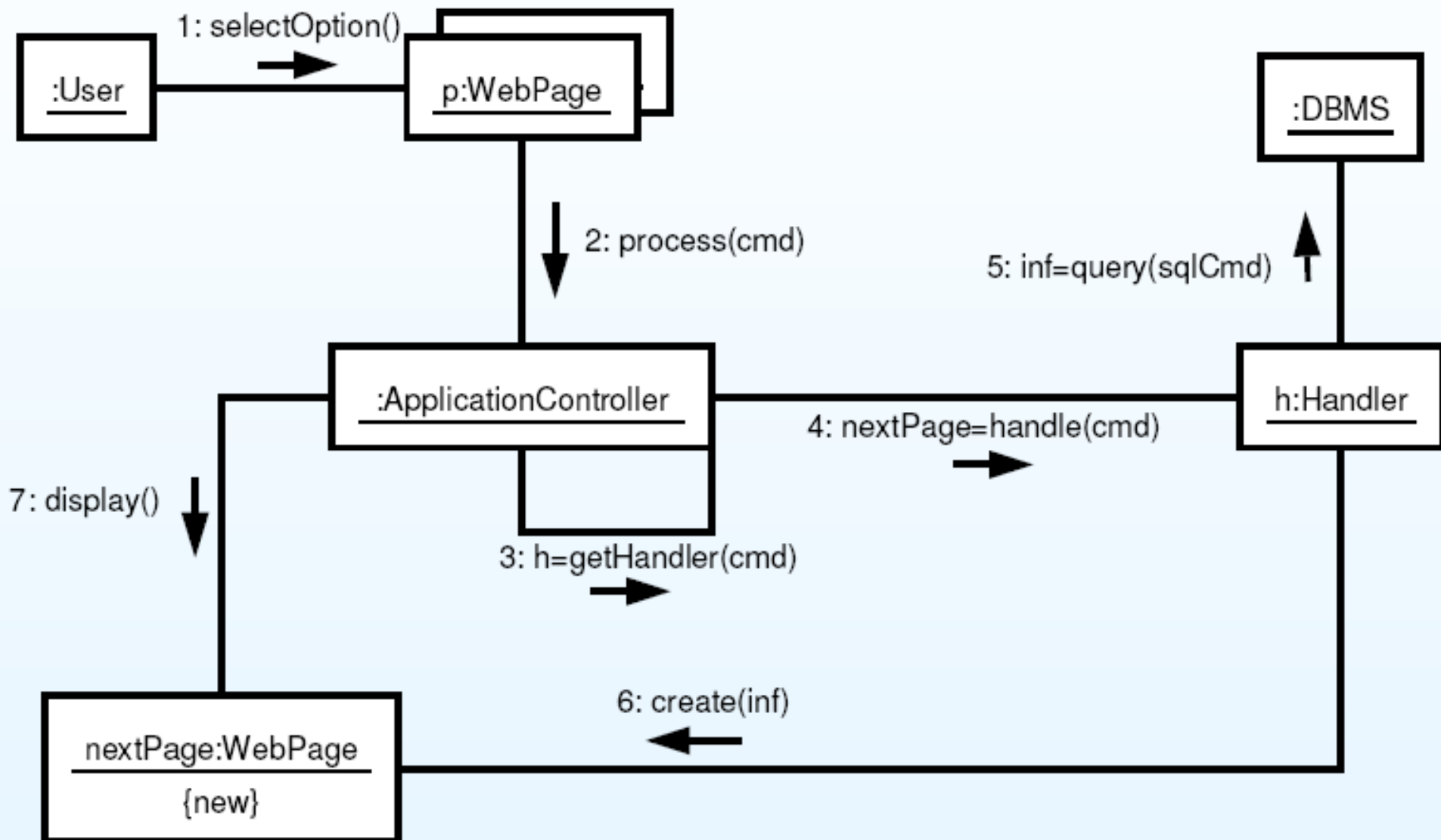
Diagramă de Colaborare

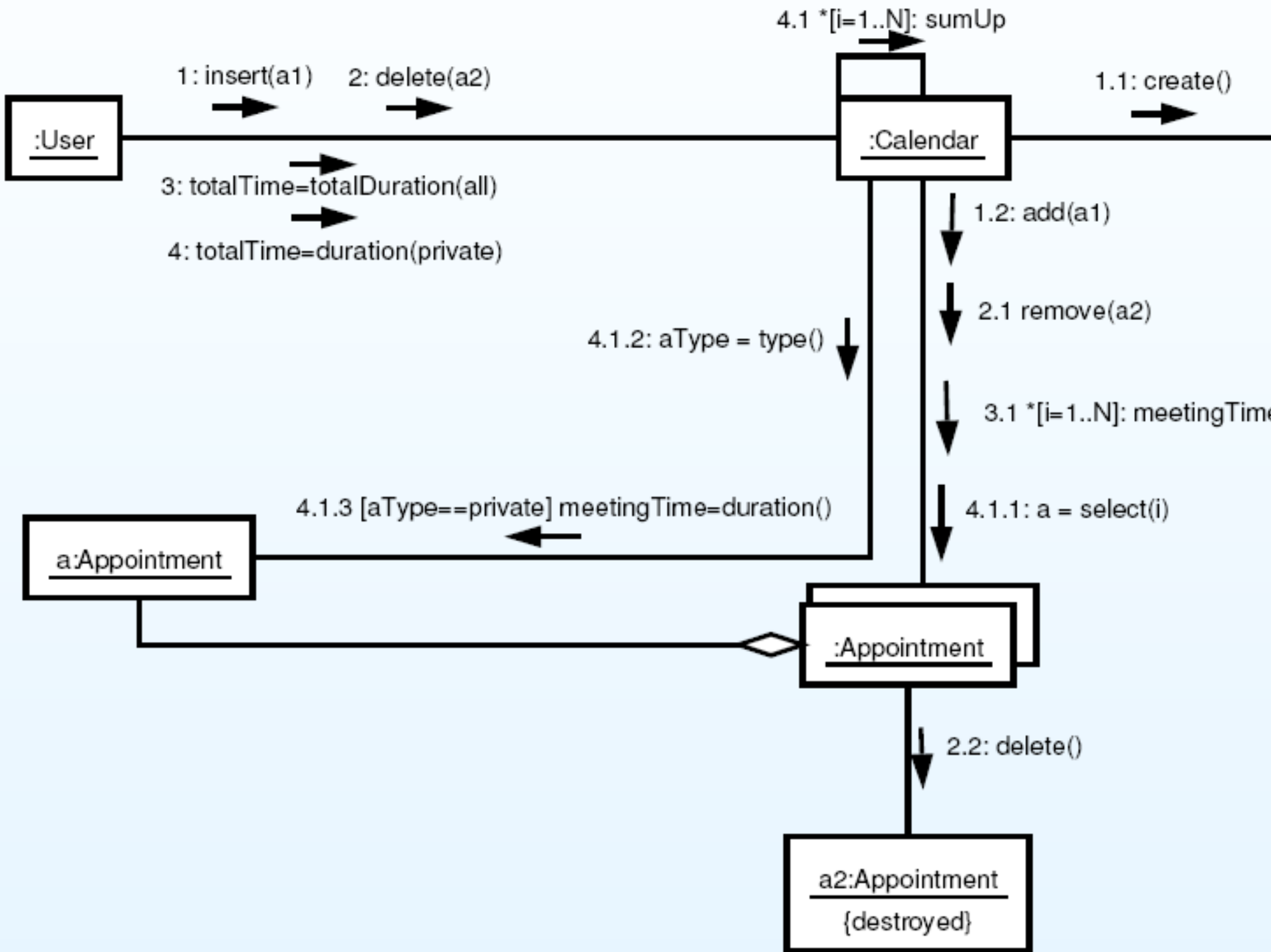
- ▶ Pune accentul pe organizarea structurală a obiectelor care participă la interacțiune
- ▶ Ilustrează mai bine ramificări complexe, iterații și comportament concurent
- ▶ Poate conține:
 - Obiecte, clase, actori
 - Legături între acestea
 - Mesaje

Tipuri de Mesaje

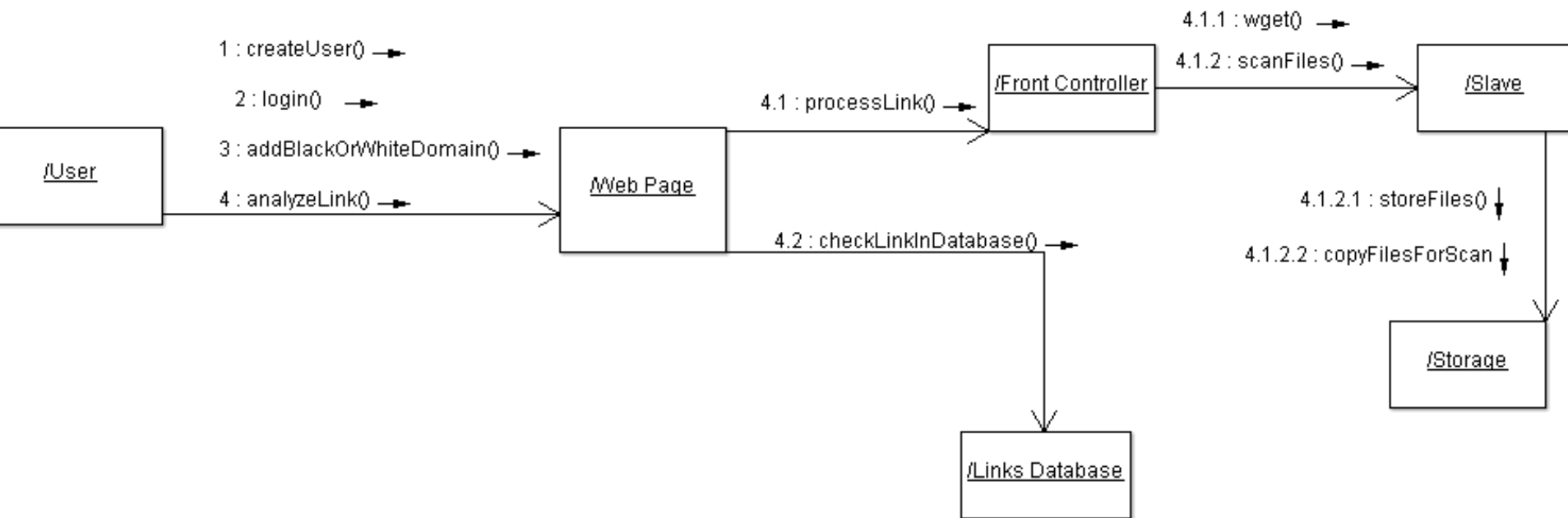
- ▶ simple
 - 2: `display(x,y)`
- ▶ subapeluri, inclusiv valoarea de retur
 - 1.3.1: `p=find(specs)`
- ▶ condiționale
 - 4: `[x<0]: invert(x,color)`
- ▶ Iterații
 - 1: `*[i=1..n]: update()`

Exemplul 1

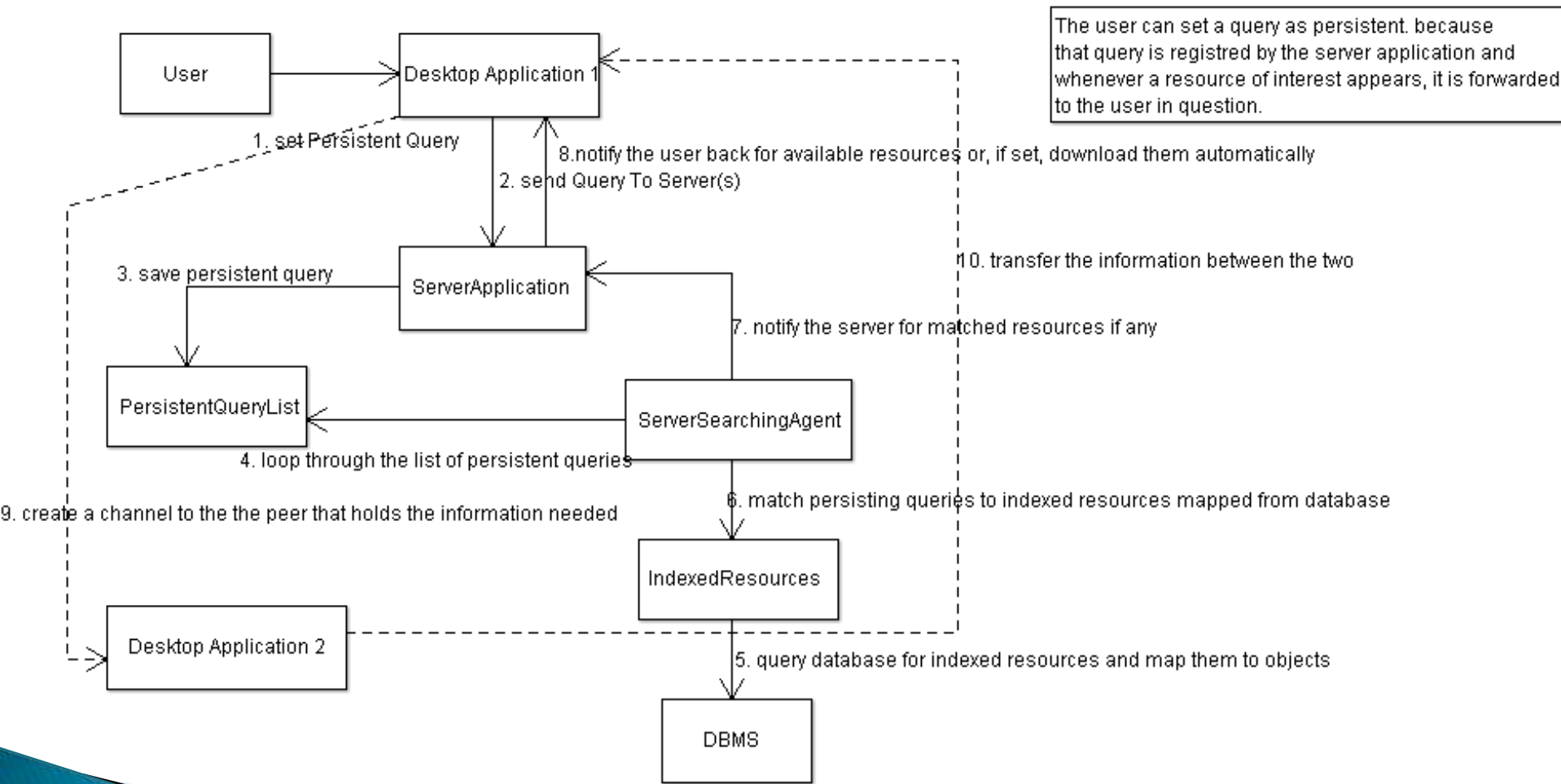




Exemplul 3



Exemplul 4



Demo 2

- ▶ Studenți bursieri
 - Diagrame de secvență
 - Diagrame de colaborare

Concluzii

- ▶ Forward engineering
- ▶ Reverse engineering
- ▶ Diagrame UML de interacțiuni
 - Diagrame de secvență
 - Diagrame de colaborare

Bibliografie

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

Links

- ▶ **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/>
- ▶ **Intelligent:** <http://www.intelligenttrd.com/>
- ▶ **Smartphones RE:** <http://www.cytraxsolutions.com/2011/01/smartphones-security-and-reverse.html>