

Diagrama de Colaborare

Collaboration Diagram

Introducere

- *Diagramele de colaborare urmăresc interacțiunile și legăturile* între un set de obiecte care colaborează.
- Atât diagramele de secvență cât și cele de colaborare vizualizează interacțiuni, dar diagrama de secvență ia în considerare timpul, pe când cea de colaborare, spațiul (contextul).
- Ca și diagramele de secvență și cele de colaborare pot fi folosite pentru a ilustra execuția unei operații, a unui caz de utilizare sau a unui scenariu de interacțiune într-un sistem.
- În acest tip de diagramă nu se pot descrie mesajele concurente și asincrone.

Utilizare

- Diagramele de colaborare pot fi folosite pentru a ilustra interacțiuni complexe între obiecte.
- Spre deosebire de diagramele de secvență, cele de colaborare ilustrează obiectele și legăturile dintre ele: o rețea de obiecte care colaborează și care în multe situații pot face mai ușoară înțelegerea interacțiunilor.

Utilizare (II)

- Pentru a ilustra o anumită interacțiune tipul diagramei se stabilește după următoarea regulă:
 - alegem *diagrama de colaborare* când o vizualizare a obiectelor și legăturilor dintre ele ne facilitează înțelegerea interacțiunilor sau când trebuie scos în evidență contextul;
 - alegem o *diagramă de secvență* când avem nevoie să vizualizăm o secvență de acțiuni sau dacă cel mai important aspect este timpul ori secvența de mesaje.

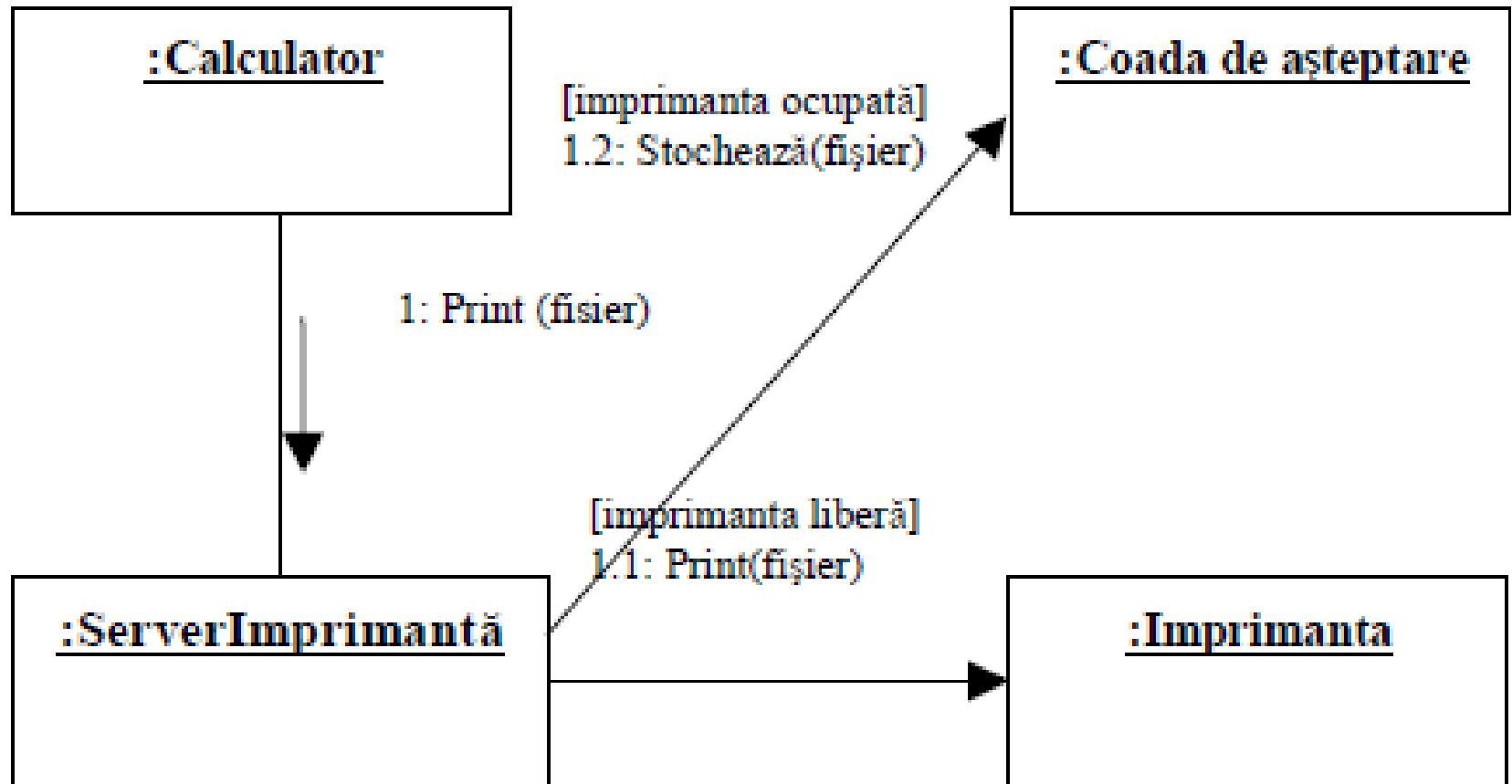
Reprezentare

- Desenarea unei diagrame de colaborare se face similar cu a unei diagrame a obiectelor.
- Mesajele vor fi reprezentate prin săgeți între obiectele implicate în mesaj și pot fi însoțite de etichete care specifică ordinea în care acestea vor fi transmise.
- De asemenea, se pot vizualiza condiții, iterații, valori returnate, precum și obiectele active care se execută concurent cu alte obiecte active.

Reprezentare (II)

- Practic, o diagramă de colaborare la nivelul instanțelor este un graf, avînd ca noduri obiectele participante la colaborare și ca arce legăturile dintre ele, însoțite de stimulii transmiși prin intermediul acestora.
- Obiectele sunt reprezentate la fel ca în diagramele de obiecte, prin dreptunghiuri ce conțin numele obiectului subliniat, dar fără a ilustra valorile atributelor.

Diagrama de colaborare: Imprimarea unui fișier



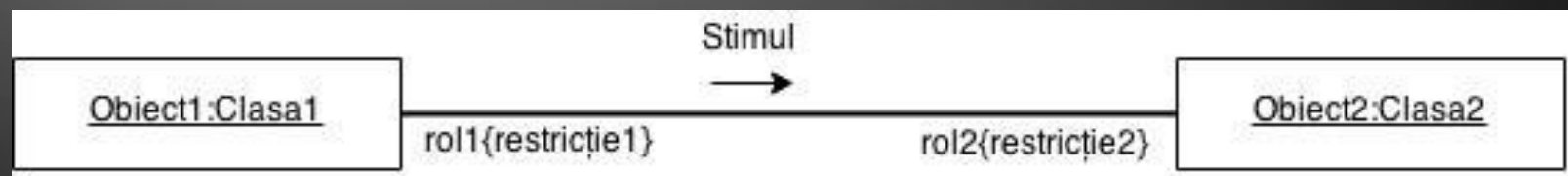
Un Actor trimite un mesaj Print unui Calculator. Acesta trimite un mesaj Print Serverului de Imprimantă, care trimite mesajul Imprimantei, dacă aceasta este liberă. În caz contrar, este stocat în Coada de așteptare.

Reprezentare (III)

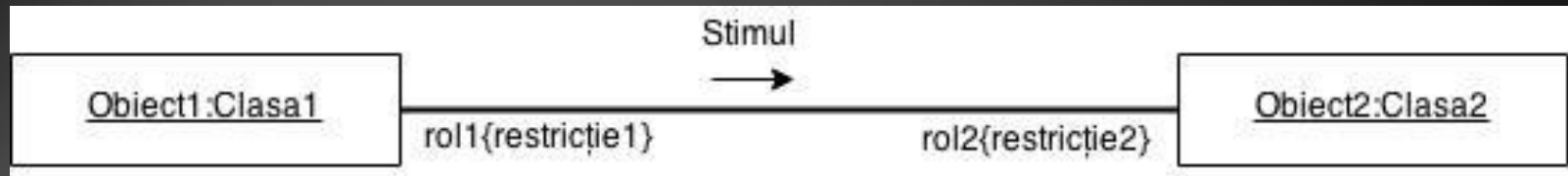
- Se poate prezenta și rolul obiectului în colaborare, folosind următoarea notație generală:

NumeObiect/'*NumeRolClasificator*': '*NumeClasificator*['.*NumeClasificator*]*

- Caracterul '*' poate apărea în colțul din dreapta sus al dreptunghiului, fiind un indicator pentru multiplicitatea obiectelor ce joacă același rol.
- Aceste elemente sunt opționale; pentru specificarea unui obiect este necesar să existe cel puțin unul dintre numele de mai sus, neapărat subliniat.



Elemente ale diagramelor de colaborare la nivelul instanțelor


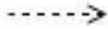



- **Legăturile** apar ca linii, având la capete, opțional, numele rolului de asociație corespunzător; pot apărea și auto-legături, marcate prin stereotipul << self >>.
- **Stimulii** se reprezintă prin săgeți mici, atașate legăturilor și indicând navigabilitatea diagramei. Acestea sunt însoțite de câte o etichetă având sintaxa:

Predecesori Condiție Secvențialitate

ValoareReturnată:=NumeStimul ListăArgumente

- **Condiționarea** unui stimul se efectuează printr-o condiție trecută între paranteze drepte, precum $[a > 1]$ StimulCondiționat.
- **Secvențialitatea** constă dintr-o secvență de numere sau nume, ce pot fi urmate de indicatori de recurență. Pentru un flux de control procedural, se efectuează o imbricare conform apelurilor imbricate de proceduri cum ar fi: $vr := \text{procedură}(\text{param})$. O execuție iterativă se specifică prin sintaxa $'*['\text{clauză-iterare}']'$, rezultând etichete precum $\text{mesaj}*[l := 1..n]$.

| | |
|---|---|
|  | Apel de procedură, însemnând un stimul sincron, la care se așteaptă răspuns |
|  | Returnarea dintr-o procedură |
|  | Stimul sincron, la care nu se așteaptă răspuns |

Tipuri de săgeți pentru reprezentarea stimulilor.

Etichete

- O etichetă asociată unui mesaj într-o diagramă de colaborare are sintaxa:

`predecesor [conditie_de_garda] expresie_secventa valoare_returnata:= semnatura`

- unde:
 - **predecesor** este o expresie pentru sincronizarea thread-urilor, ceea ce înseamnă că mesajele, pentru specificarea numerelor de secvență trebuie să fie realizate și gestionate înainte ca mesajul curent să fie trimis (execuția este continuă). În lista, numerele de secvență sunt separate prin virgula.
 - **conditia_de_garda** este prinsă între paranteze drepte.
 - **numărul de secvență** specifică ordinea mesajelor. Mesajul 1 va începe întotdeauna secvența de mesaje; Primul tratat, când începe gestionarea mesajului 1 este mesajul 1.1, apoi 1.2, respectiv 1.2.1., ș.a.m.d.

Recurența

- Recurența reprezintă o execuție condițională sau iterativă. Sunt două posibilități:
- 1. ***[clauza_de_iterare]** – folosită pentru specificarea unei iterații (execuția repetată). Între paranteze drepte este prinsă condiția de iterare. De exemplu: `[i: =1..n]`.
- Presupunem că avem o etichetă de mesaj care conține o iterație; aceasta poate fi:

`*[x= 1..10] : executaCeva()`

- 2. **[clauza_conditie]** – este folosită de obicei pentru specificarea ramurilor, și nu pentru condiții de gardă.
- De exemplu, `[x>=0]` și `[x<0]` sunt două *clauze-condiții* care pot fi folosite pentru ramificare, când numai una din condiții este adevărată, așadar numai una din ramuri va fi executată.

Etichete (II)

- **Valoarea returnată** va fi asignată unei semnături de mesaj. O semnătură este compusă din numele mesajului și o listă de argumente.

Valoarea returnată este valoarea obținută ca rezultat al apelului operației (un mesaj).

De exemplu:

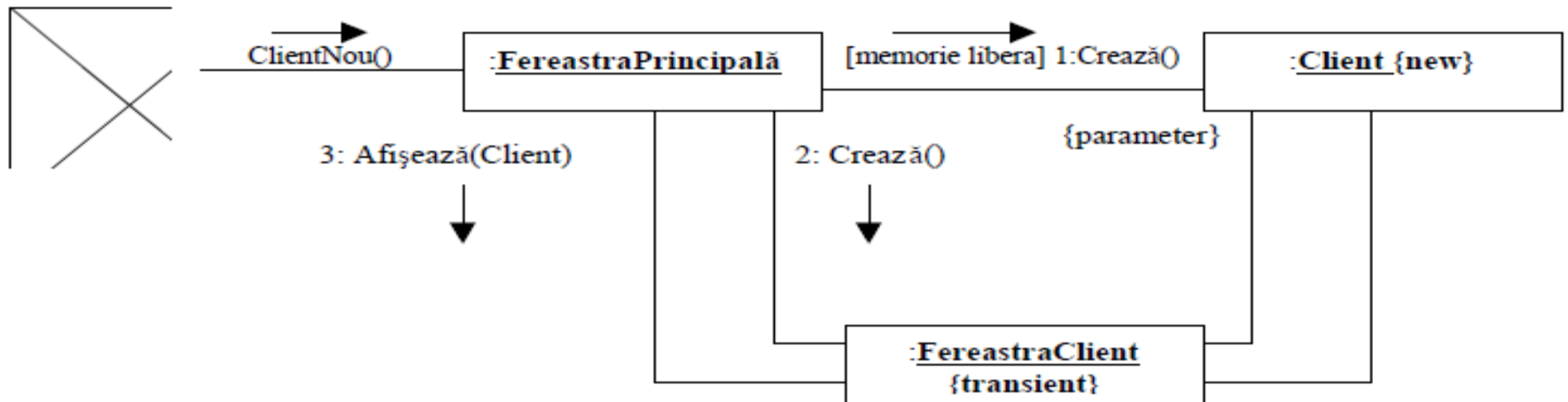
1: $x := \text{calc}(n)$.

Exemple de etichete de mesaj:

- 1: display()
- [mode = display] 1.2.3.7:redraw()
- 2 *[n:=1..z]: prim :=nextPrim (prim)
- 3.1 [x<0]: foo()
- 3.2 [x>=0]: bar()
- 1.1 a,1.1 b/1.2: continue()

Timpul de viață al unui obiect

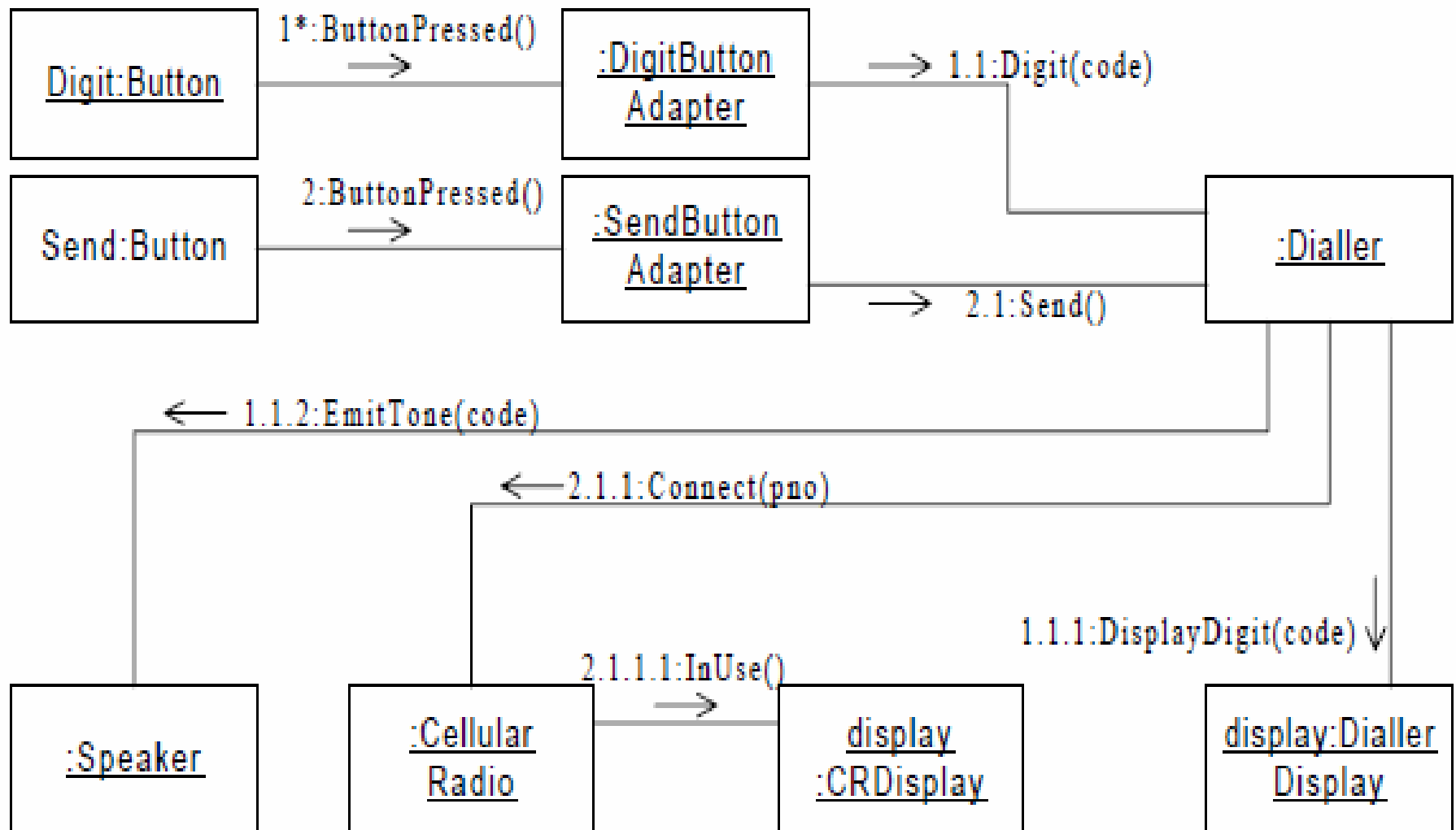
- Obiectele create pe parcursul colaborării sunt descrise cu `{new}`, obiectele distruse cu `{destroy}`, iar obiectele care sunt atât create cât și distruse în aceeași colaborare sunt descrise cu `{transient}`, care este echivalentul unui `{new} {destroy}`.



3.1: Actualizare(data)

Obiectul FereastraPrincipală recepționează mesajul `ClientNou()` și, dacă este memorie liberă, crează un obiect Client. Apoi este creat un obiect FereastraClient și obiectul Client este transmis obiectului FereastraClient, în care se vor putea modifica datele clientului.

Exemplu



Temă de laborator

Elaborați diagrama de colaborare folosindu-vă de diagrama de secvență realizată la laboratorul anterior.