Laborator 6

Diagrama de colaborare

Diagrama de colaborare este o diagramă de interacțiuni care pune accentul pe organizarea structurală a obiectelor care participă la interacțiune.

Diagrama de colaborare poate conține:

- obiecte;
- legături între obiecte;
- mesajele prin care obiectele comunică.

Diagramele de colaborare au multe asemănări cu diagramele de secvență, exprimând aceleaşi informații dar într-un alt format. Pot fi create la nivele diverse de detaliu şi în diferite stadii de dezvoltare a procesului software. Deoarece au un conținut similar, pot fi folosite pentru generarea diagramelor de secvență şi viceversa.

Diferența semnificativă față de diagrama de secvență este aceea că diagrama de colaborare arată explicit legăturile dintre obiecte. De asemenea, la diagrama de colaborare timpul nu are o dimensiune explicită. Din acest motiv, ordinea în care sunt trimise mesajele este reprezentată prin numere de secvență.

Mesajele dintr-o diagramă de colaborare sunt reprezentate de un set de simboluri care sunt asemănătoare celor utilizate în diagrama de secvență, dar cu câteva elemente adiționale pentru a marca secvențierea şi recurența.

Notă. Termenii care vor apărea între paranteze pătrate sunt opționali iar termenii care vor apărea între acolade pot să nu apară sau să apară de mai multe ori.

Sintaxa unui mesaj este următoarea:

```
[predecesor] [condiție_gardă] expresie_secvență [valoare_întoarsă
             ':='] nume_mesaj '('[listă_argumente]')'
```

unde

- *predecesor* – listă a numerelor de secvență a mesajelor care trebuie trimise înainte de trimiterea mesajului curent (permite sincronizarea trimiterii mesajelor); permite specificarea detaliată a căilor ramificațiilor.

  Sintaxa *predecesorului* este următoarea:

  ```
  număr_secvență { ',' număr_secvență } '/'
  ```

  '/' – marchează sfârşitul listei şi se include doar dacă este precizat explicit predecesorul.

- *condiție_gardă* – expresie booleană care permite condiționarea transmiterii mesajului (se scrie în OCL – Object Constraint Language) şi se poate utiliza pentru reprezentarea sincronizării diferitelor fluxuri de control.

- *expresie_secvență* – listă de întregi separați prin caracterul `'.'`, urmată opțional de un *nume* (o singură literă), un termen recurență şi terminată de caracterul `':'`.

  Sintaxa *expresiei_ secvență* este următoarea:

  ```
  întreg { '.' întreg } [nume] [recurență] ':'
  ```

Se observă că numerotarea este asemănătoare celei utilizate la numerotarea capitolelor şi paragrafelor într-un document.

întreg – precizează ordinea mesajului; poate fi folosit într-o construcție de tip buclă sau ramificație.

Exemplu: mesajul 5.1.3 se transmite dupa 5.1.2 şi ambele se transmit după activarea mesajului 5.1.

nume – se foloseşte pentru a diferenția două mesaje concurente când acestea au acelaşi număr de secvență.

Exemplu: mesajele 3.2.1a şi 3.2.1b se transmit simultan în cadrul activării mesajului 3.2.

recurența – permite specificarea modului de transmitere a mesajelor:

- secvențial - `'*''['clauză_iterație']'`
- paralel - `'||''['clauză_iterație']'`
- ramificație - `'['clauză_condiție']'`

În tabelul următor se prezintă câteva exemple de tipuri de mesaje:

Tipuri de mesaje:	**Exemple**
mesaj simplu	`4: adaugă ReclamaNoua ()`
subapeluri cu valoarea întoarsă *Valoarea întoarsă este plasată în variabila nume*	`3.1.2: nume:= extrageNume ()`
mesaj condițional *mesajul este trimis doar dacă este adevărată condiția* `[balanță > 0]`	`[balanță > 0] 5: debit (sumă)`
sincronizare cu alte mesaje *mesajul* `4: playVideo` *este trimis doar după ce mesajele concurente* `3.1a` *şi* `3.1b` *sunt complete*	`3.1a, 3.1b / 4: playVideo()`
iterații	`[i = 1..n] update ()`

În figura 1 este prezentată diagrama de colaborare corespunzătoare primei diagrame de secvență din laboratorul anterior.

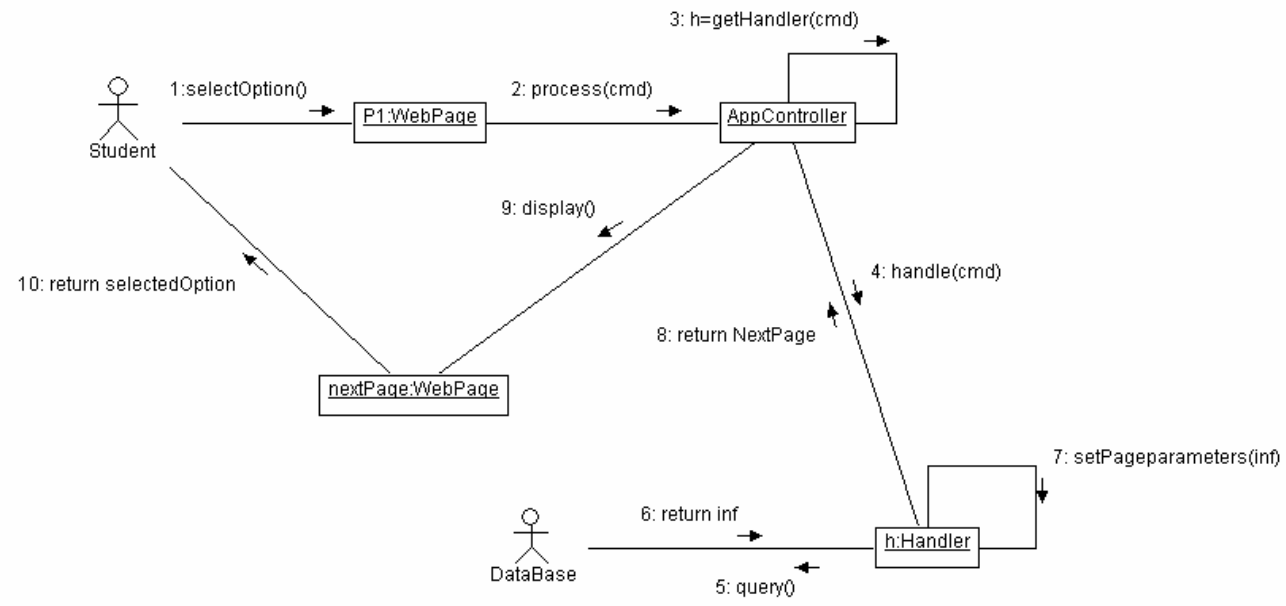

Figura 1. Exemplu de diagramă de colaborare

Există mai multe posibilități de interacțiune pentru un use case particular. Acestea se datorează alocările posibile diferite a responsabilităților. Spre exemplu, interacțiunile din figura 2 pot avea trăsături nedorite. Mesajul `extrageDetaliiCampanie` trimis de `Client` obiectului `Campanie` necesită ca obiectul `Client` să returneze aceste detalii obiectului `AdaugaReclama`. Dacă detaliile despre campanie includ doar numele campaniei atunci un volum relativ mic de date este pasat de la `Campanie` la `Client` şi apoi la `AdaugaReclama`. Acest fapt poate fi acceptabil.

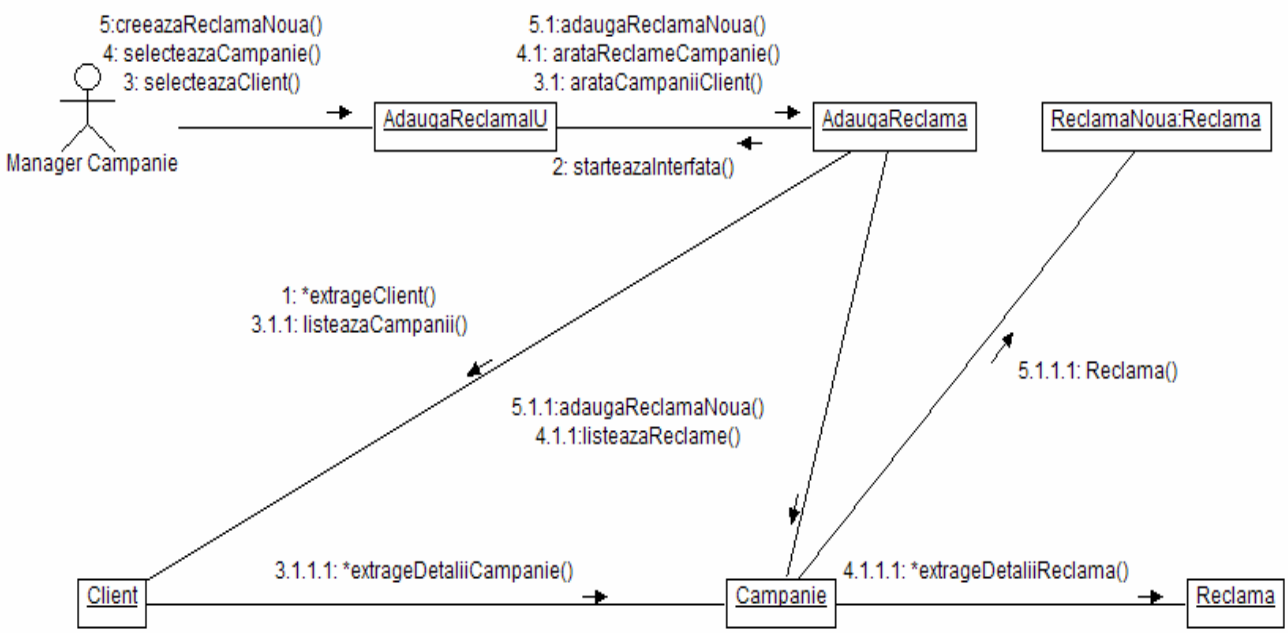

Figura 2. Diagrama de colaborare pentru use case-ul `Adaugă o reclamă nouă unei campanii`

Pe de altă parte, dacă detaliile despre campanie includ data de start, de terminare şi bugetul campaniei, atunci prin `Client` se pasează mai mult de o dată. În acest caz obiectul `Client` este acum responsabil pentru furnizarea unor date semnificative pentru campanii în loc de obiectul `Campanie`. S-ar putea deci transfera date direct de la `Campanie` la `AdaugaReclama`. Această posibilitate este prezentată în figura 3, în care `AdaugaReclama` preia responsabilitatea de a extrage detalii despre campanie direct de la obiectul `Campanie`. În această interacțiune, obiectul `Client` este responsabil doar de furnizarea unei liste de campanii obiectului `AdaugăReclamă`.

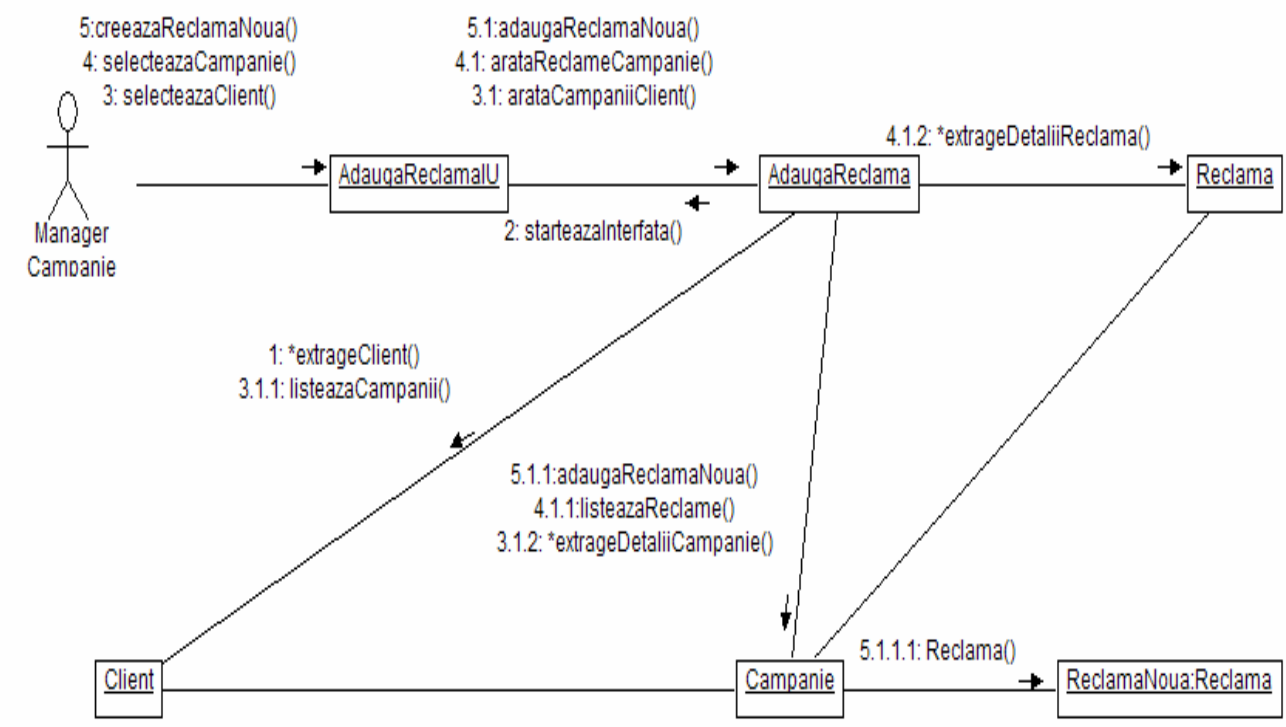

Figura 3. Diagramă de colaborare alternativă pentru use case-ul `Adaugă o reclamă nouă unei campanii`

În figura 4 este prezentată o diagramă de colaborare care prezintă interacțiunile pentru o singură operație – `verificăBugetCampanie()` – care este una din operațiile din diagramele din figurile 2 şi 3.

Diagramele de colaborare sunt preferate de unii dezvoltatori diagramelor de secvență deoarece interacțiunile între obiecte pot fi translate uşor în diagramele de clase datorită vizibilității legăturilor între obiecte.

Probleme propuse

Pentru fiecare din problemele de mai jos să se realizeze diagramele de colaborare:

1. Automat cafea (alegere tip cafea, introducere moneda, eliberare rest, preluare produs, etc)
2. ATM (verificare PIN, vizualizare suma din contul personal, extragere, tipărire chitanța etc.)
3. Ceas electronic (afişare ora curentă / data curentă, modificare oră / dată, cronometru etc.)

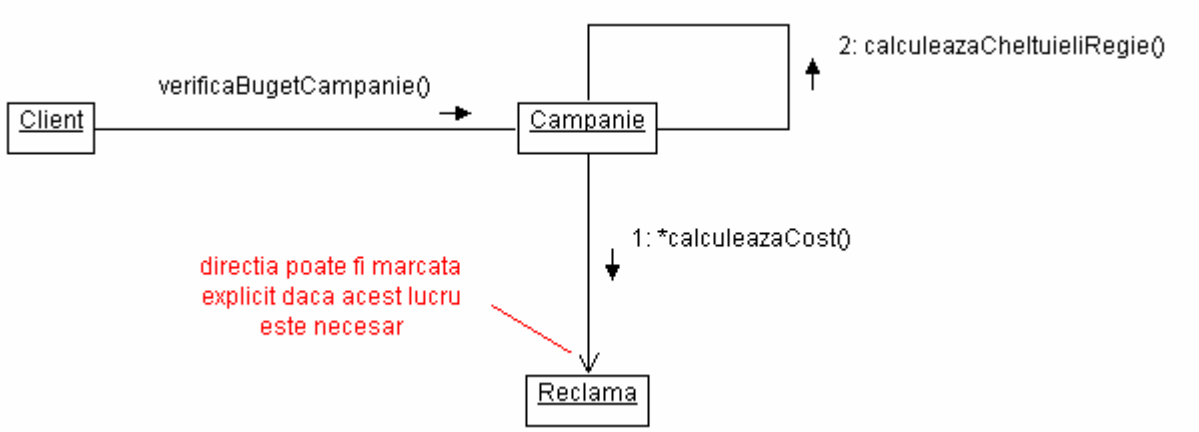

Figura 4. Diagrama de colaborare pentru operația `verificaBugetCampanie()`