Inginerie Software 2024
Laboratorul 2
Modelarea sistemului în context: funcții. Cazuri de utilizare (Use Cases) *

Mădălina Erașcu, Alexandru Munteanu, Cristian Pal, Ionica Puiu

February 2024

Obiective:

1. Utilizarea diagramei cazurilor de utilizare în diferite contexte.

1 Introducere

Un **sistem software** oferă un **set de funcții**. Acestea sunt utilizate de **entități din contextul** său (actori) care **interacționează** cu sistemul.

Primul nivel de modelare se referă la:

- funcțiile oferite de sistem
- interacțiunile entităților din context cu sistemul pentru a obține aceste funcții. Abordarea se bazează pe **cazuri de utilizare**.

Un **caz de utilizare** defineşte interacţiunile dintre actorii externi şi sistem în scopul realizării unui anumit obiectiv.

Actor specifică un rol [1] jucat de o persoană sau o altă entitate (un alt sistem, un dispozitiv) atunci când interacţionează cu sistemul.

"Use case is a behaviourally related sequence of transactions performed by an actor in a dialogue with the system to provide some measurable value to the actor" [1].

Actorii există în afara sistemului studiat şi participă la o secvenţă de activităţi în *dialog cu sistemul*, în scopul îndeplinirii unui anumit *obiectiv*. *Modelul sistemului* conține:

- Diagrama cazurilor de utilizare
- Detaliile pentru fiecare caz de utilizare

*Bazat pe resursele de laborator ale Conf. Dr. Cristina Mândruță

[1] Aceeaşi persoană poate utiliza sistemul ca actori diferiţi deoarece aceasta poate juca roluri diferite la momente de timp diferite.

Remarcă. *Analiza cazurilor de utilizare este forma primară de culegere a cerințelor de utilizare pentru un nou program software sau sarcină de îndeplinit.*

Obiectivele primare ale analizei cazurilor de utilizare sunt:

- *proiectarea unui sistem din pespectiva utilizatorului*
- *comunicare comportamentului sistemului în termenii utilizatorului*
- *specificarea tuturor interacțiunilor vizibile din exterior.*

2 Diagrama Cazurilor de Utilizare

- Scop: **Ce** trebuie să facă sistemul.
- Descrie **funcţionalitatea** sistemului din punctul de vedere al utilizatorului.
- Nivel superior de abstractizare.
- Modelare centrată pe obiectiv: scopul (**ce** face sistemul) nu soluția (**cum**).

2.1 Elemente de modelare

Actor – entitate ce interacţionează cu sistemul; rol jucat de o persoană, de un alt sistem sau de un dispozitiv hardware.
Sistem – sistemul modelat.
Caz de utilizare (Use Case) – un serviciu pe care sistemul ştie să-l furnizeze; funcţie cheie a sistemului.
Tipuri de relaţii:

- **asociere** – interacţiune între actor şi caz de utilizare.
- ≪include≫ – între două cazuri de utilizare; cazul de utilizare inclus este un caz de utilizare reutilizabil care este incorporat necondiţionat în executarea cazului de utilizare ce îl include; cazul de utilizare apelant decide când şi de ce utilizează cazul de utilizare inclus.
- ≪extend≫ – între două cazuri de utilizare; cazul de utilizare care extinde este un caz de utilizare reutilizabil care întrerupe condiţionat execuţia cazului de utilizare extins pentru a-i extinde funcţia; cazul de utilizare care extinde decide când va fi folosit.
- **generalizare** – relaţie de moștenire între actori sau între cazuri de utilizare.

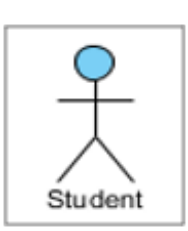
2.2 Notații

2.2.1 Actor

Vezi Figura 1.

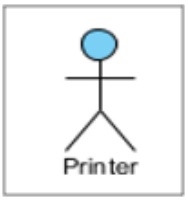
2.2.2 Caz de utilizare (Use case)

Vezi Figura 2.
Numele cazului de utilizare trebuie să înceapă cu un verb!



(a) O persoană

(b) Un alt sistem



(c) Un dispozitiv extern

Figure 1: Tipuri de Actori

Figure 2: Caz de utilizare

2.2.3 Relaţii

Pentru un sumar al relațiilor, consultați Tabelul 1.

1. Generalizare (vz. Figura 3)

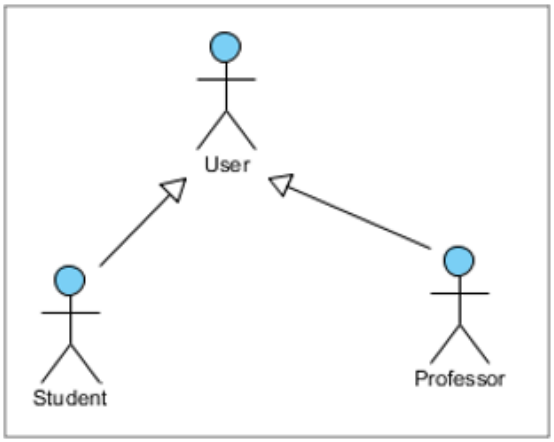


Figure 3: Exemplu de relație de generalizare: Actorii "Student" și "Profesor" sunt specializări ale actorului "User" care este mai general.

2. Incluziune (≪include≫) (vz. Figura 4)
3. Extindere (≪extend≫) (vz. Figura 5)

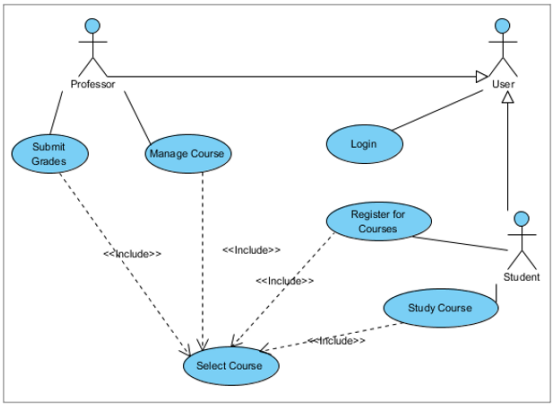


Figure 4: Exemplu de relație de incluziune: Cazul de utilizare "Select Course" va fi executat de fiecare datã când va fi lansat "Register for Courses" sau "Study Course".

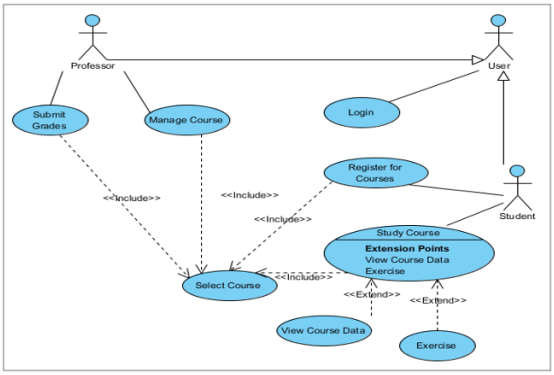


Figure 5: Exemplu de relație de extindere: Cazul de utilizare "View Course Data" va fi executat doar în punctul de extensie "View Course Data" al cazului de utilizare "Study Course". Similar pentru cazul de utilizare "Exercise".

2.3 Marcarea limitelor sistemului (opţional)

Cazurile de utilizare pot fi incluse într-un dreptunghi care va delimita sistemul de contextul său. Tot ce e conţinut în acest dreptunghi reprezintă funcţionalitatea aflată în zona sistemului şi tot ce este în afara acestuia se află în afara sistemului.

Remarcă. *Reprezentarea limitelor poate fi utilizată şi pentru a identifica ce cazuri de utilizare vor fi livrate în fiecare versiune majoră a sistemului.*

3 Reguli

- Un caz de utilizare definește un serviciu, nu secvenţe de operaţii.

include	**extend**
Extinde comportamentul cazului de utilizare de bază.	Extinde comportamentul cazului de utilizare de bază.
Cazul de utilizare inclus este folosit întotdeauna pentru a extinde cazul de utilizare în curs de execuţie.	Cazul de utilizare care extinde ar putea fi folosit (uneori) pentru a extinde cazul de de utilizare în curs de execuţie.
Cazul de utilizare în curs de execuţie decide când apelează cazul de utilizare inclus. Cazul de utilizare inclus nu este conştient de cazul de utilizare de bază.	Cazul de utilizare care extinde decide când se va insera în execuţia cazului de utilizare de bază. Cazul de utilizare de bază nu este conştient de cazul de utilizare care extinde.
Săgeata este desenată către cazul de utilizare inclus.	Săgeata este desenată către cazul de utilizare extins.

Table 1: Sumar al relațiilor

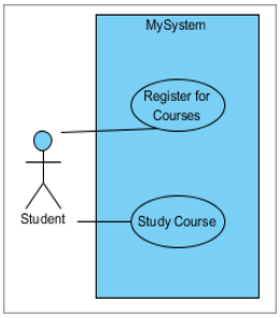


Figure 6: Delimitare sistem de context

- Toţi actorii comunică cu un singur sistem.
- Actorii nu comunică direct între ei.
- Trebuie identificate toate funcțiile de nivel înalt ale sistemului, adică ceea ce sistemul ştie să facă pentru actorii săi.

Remarcă. *Explicați erorile din Figurile 7 and 8.*

4 Studiu

1. Studiaţi crearea de diagrame de cazuri de utilizare de la `https://www.visual-paradigm.com/support/documents/vpuserguide/94/2575/6362_drawinguseca.html`.

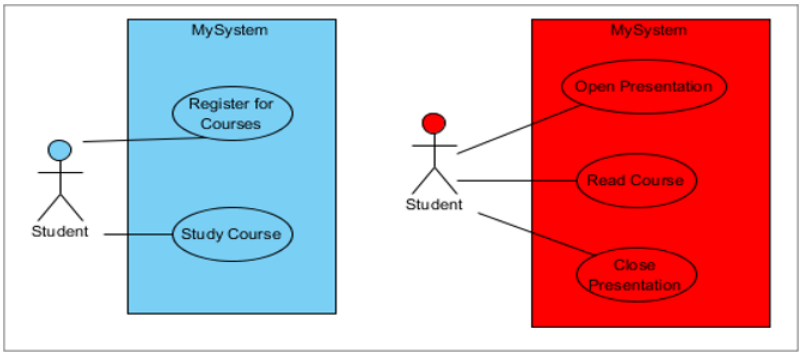


Figure 7: Exemplu 1. Corect - Gresit

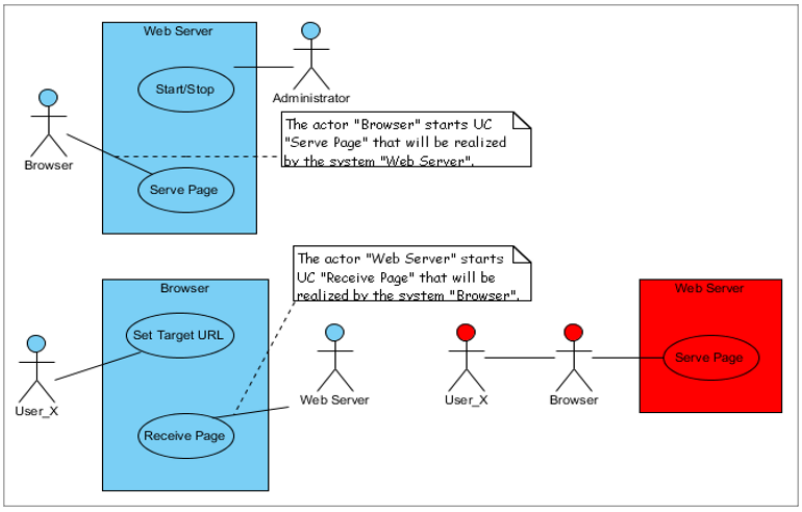


Figure 8: Exemplu 2. Corect - Gresit

5 Temă

1. Creaţi o diagramă UC pe baza următoarei descrieri. Dezvoltăm un sistem cu care profesorii pot înregistra şi actualiza notele studenţilor. Profesorii trebuie să poată distribui rapoarte cu notele. Iată lista completă a cerinţelor sistemului:
   - Un profesor poate înregistra note. De câte ori sunt înregistrate note, acestea sunt salvate pe disc.
   - Un profesor poate actualiza note. De cîte ori sunt actualizate note, nota existentă este încărcată. După modificare noua notă este salvată pe disc.
   - Profesorul, secretara şi studentul pot vizualiza note.
   - Pentru a vizualiza note, solicitantul trebuie să se conecteze la sistem.

Dacă eşuează conectarea, solicitantul trebuie să se re-autentifice indicând nume şi parolă.
- Un student cu taxă este un tip de student.
- O secretară poate genera rapoarte cu notele.
- Un profesor poate distribui rapoarte cu notele.

Cerințe
- Identificaţi actorii.
- Există actori care sunt specializări ale altor actori mai generali? Dacă da, identificaţi care este actorul general şi care este actorul specializat. Ce tip de relaţie trebuie reprezentată între actorul general şi specializarea sa?
- Identificaţi cazurile de utilizare.
- Există cazuri de utilizare folosite întotdeauna de alte cazuri de utilizare? Dacă da, ce tip de relaţie există între acestea? Identificaţi cazurile de utilizare folosite întotdeuna şi cazurile de utilizare ce le folosesc.
- Există cazuri de utilizare folosite uneori de către un alt caz de utilizare? Dacă da, ce tip de relaţie există între acestea? Identificaţi cazurile de utilizare folosite uneori şi cazurile de utilizare ce le folosesc.
- Desenaţi diagrama UC corespunzătoare, incluzând toţi actorii, cazurile de utilizare, şi relaţiile. Atenţie la folosirea notaţiilor corespunzătoare şi a etichetelor pentru actori, cazuri de utilizare şi relaţii.

2. Desenaţi o diagramă UC pentru următoarea aplicaţie. O bibliotecă conţine cărţi şi jurnale. Se cere dezvoltarea unui sistem softwarea pentru împrumutul de cărţi. Pentru a împrumuta o carte, clientul trebuie să fie membru al bibliotecii. Există o limită a numărului de cărţi ce pot fi simultan împrumutate de un membru al bibliotecii. Biblioteca poate deţine mai multe exemplare ale unei anumite cărţi. O carte se poate rezerva. Unele cărţi pot fi împrumutate doar pe termen scurt. Alte cărţi pot fi împrumutate pentru 3 săptămâni. Utilizatorii pot extinde înprumuturile.

3. Fie următoarea descriere a cerințelor unei aplicații software pentru o firmă de închiriere mașini. Firma are mai multe oficii de unde se pot închiria mașinile. Identificaţi actorii şi utilizaţi descrierile de mai jos ca bază pentru definirea cazurilor de utilizare. Desenaţi un model al cazurilor de utilizare ilustrând toate relaţiile dintre cazurile de utilizare. Reprezentaţi apoi în VP for UML diagrama cazurilor de utilizare.

   REZERVARE
   Clientul accesează site-ul firmei cu scopul de a face o rezervare. Pe site i se va afișa un formular în care va trebui să indice data iniţială şi data finală, vehicolul preferat şi oficiul de închiriere de unde doreşte să închirieze maşina. După ce datele sunt trimise sistemul verifică dacă este disponibil un vehicol corespunzător solicitării clientului. Dacă vehicolul solicitat este

disponibil, atunci sistemul calculează prețul și îl afișează clientului. Clientul acceptă prețul afișat. Sistemul înregistrează rezervarea iar clientul primește un număr de închiriere. Sistemul crează un acord de închiriere, incluzând numărul de închiriere, perioada închirierii, tipul vehicolului şi oficiul de închiriere.

VERIFICARE DISPONIBILITATE

Sistemul verifică disponibilitatea pentru a vedea dacă este disponibil un vehicol de un anumit tip, la un anumit oficiu de închiriere, pentru o perioadă dată de timp. Pentru fiecare vehicol sistemul cunoaște intervalale de timp când este disponibil şi când nu. Dacă este disponibil, vehicolul este rezervat pentru perioada solicitată.

INIŢIERE ÎNCHIRIERE

Clientul ajunge la oficiul de închiriere şi indică angajatului numărul de închiriere. Angajatul introduce numărul de închiriere în sistem. Sistemul caută acordul de închiriere corespunzător și îi afișează pentru a fi discutat cu clientul. Dacă clientul acceptă acordul de închiriere atunci acesta este imprimat pentru a fi semnat de către client. Apoi sistemul afișează o listă de opțiuni de asigurare existente. Clientul indica opțiunea preferată. Angajatul introduce în sistem preferința clientului. Sistemul imprimă formularul poliței de asigurare pentru a fi semnată de către client și atașată la acordul semnat.

PROCESARE RETURNARE VEHICOL

Clientul înregistrează kilometrajul şi nivelul de combustibil şi le indică angajatului care le introduce în sistem. Sistemul calculează cantitatea de combustibil consumată de client și o adaugă la contul închirierii din acordul de închiriere. Contul închirierii, care include costurile totale ale închirierii, este afișat de sistem și verificat de către client. Clientul plăteşte costurile închirierii. Angajatul înregistrează în sistem faptul că plata a fost făcută.

CREARE RAPOARTE MANAGEMENT

Sistemul poate genera mai multe tipuri de rapoarte. Managerul firmei selectează un tip de raport. Sistemul generează și afișează raportul solicitat. Dacă managerul solicită imprimare, sistemul imprimă raportul.

OPȚIONAL: O EXTENSIE ULTERIOARĂ

Extindeţi proiectul prin adăugarea de caracteristici suplimentare suport pentru închirieri regulate de către firme client. În acest caz vor fi disponibile zilnic un număr precizat de vehicole pentru angajaţii companiei client, la oficiile de închiriere precizate. Firma client poate nominaliza şi oferi o listă cu angajaţii autorizaţi ce pot ridica vehicolele. Firmei client i se va prezenta lunar un cont de plată. Pentru aceasta va trebui ca mai întâi să scrieţi un nou caz de utilizare şi/sau să adaptaţi cazurile de utilizare existente.

References

[1] Ivar Jacobson, Maria Ericsson, and Agneta Jacobson. *The object advantage: business process reengineering with object technology.* ACM Press/Addison-Wesley Publishing Co., 1994.