Kompozíciós mechanizmusok

forrás:

- A Taxonomy of Software Composition Mechanisms. K.-K. Lau and T. Rana. In *Proceedings of Thirty-sixth Euromicro Conference on Software Engineering and Advanced Applications*, pages 102-110, IEEE, 2010.

Szoftver kompozíció

- szoftver konstrukciók összeállítása nagyobb kompozit szoftver konstrukciókká
  - komponensek
  - újrafelhasználás
  - szisztematikus szoftver konstrukció
- kompozíciós mechanizmus
  - szoftver konstrukciók és a köztük lehetséges interakciók
  - adattípus–összetett adattípus, kód–szkript, objektumok–üzenetküldés, szoftver architektúra–portok, …
- szisztematikus és automatizálható kell

Szoftver kompozíció nézetei

- különféle kompozíciós egységek lehetségesek
- itt most a viselkedést meghatározó szoftver komponensek jelentik a kompozíciós egységet (adattípusok, adatstruktúrák nem)
- kompozíciós mechanizmus viselkedésekből összetett viselkedést hoz létre
- a kompozíció különböző nézetei
  - programozási nézet
  - konstrukciós nézet
  - komponens alapú fejlesztési (CBD) nézet

Programozási nézet

- amit a programozó csinál, amikor kódrészletekből összeállítja a programot vagy alkalmazást
- kompozíciós egység: programnyelvi konstrukció
  - függvények, eljárások, osztályok, aspektusok, …
- kompozíciós mechanizmus: programnyelv által megengedett szekvencia
- háttérbe szorul az újrafelhasználás
- „programming in the small”

Konstrukciós nézet

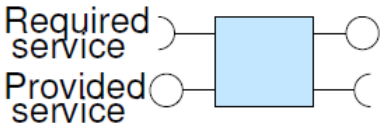
- alkalmazások készítése komponensek csatolókon keresztüli összekapcsolásával
- motivációja a szisztematikus konstrukció
- kompozíciós egység: csatolókkal rendelkező komponensek
  - modulok, Java Bean, …
- kompozíciós mechanizmus: szkript nyelvek
- a tervezés eredménye a szoftver architektúra
  - komponensek és a köztük levő kapcsolatok
- háttérbe szorul a létező egységek készlete
- „programming in the large”

CBD nézet

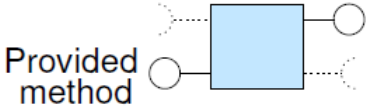
- a konstrukciós nézetet kiegészíti komponensmodellel és létező elemek újrafelhasználásával
- motiváció: szisztematikus újrafelhasználása már létező szoftverkonstrukcióknak
- kompozíciós egységek: objektumok, architektúrális egységek, befoglalt komponensek, …
- kompozíciós mechanizmus az egységek sajátosságai szerint (lásd. köv.)

CBD nézet

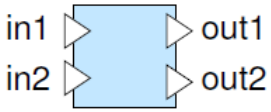
- objektumoknak van szolgáltatott interfészük, de nincs (explicit) elvárt interfészük
- architekturális egységeknek van ki- és bemeneti portjuk (szolgáltatott és elvárt interfész)
- befoglalt komponensek(*) számítást (adat és vezérlés) foglalnak magukban, csak szolgáltatott interfészük van és nincsen elvárt interfészük (nem hívnak mást)



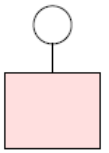
(a) A generic component



(b) An object



(c) An architectural unit



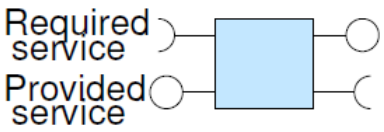
(d) An encapsulated component

(*) lásd később

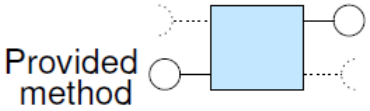
CBD nézet

- az általános komponensek a szolgáltatott és elvárt interfészek illesztésével kapcsolódnak
- objektumok nem tudnak így kapcsolódni, mert nincs elvárt interfészük, ezért metódus hívással kapcsolódnak
- architekturális egységek a kompatibilis portokkal kapcsolódnak
- befoglalt komponensek közvetlenül nem tudnak kapcsolódni, kell nekik exogén kompozíciós konnektor(*)

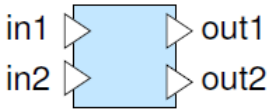
(*) lásd következő slide



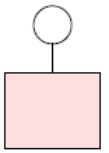
(a) A generic component



(b) An object



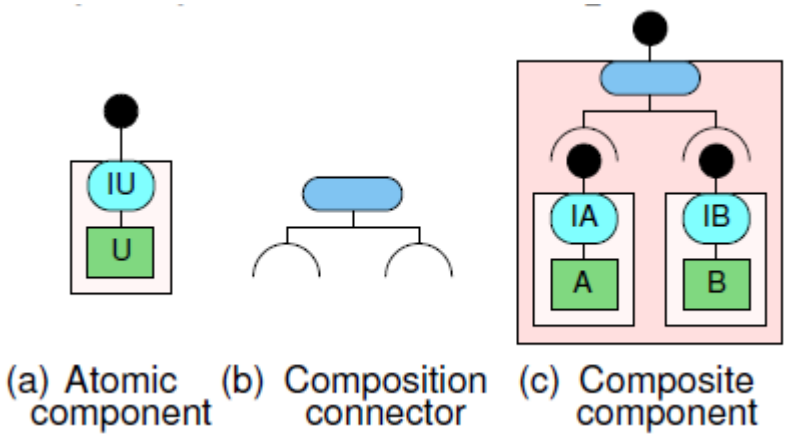
(c) An architectural unit



(d) An encapsulated component

CBD nézet
exogén kompozíció

- a komponensek teljesen befoglaltak, a komponensen kívülre nem megy a vezérlés
- atomi komponens
  - számítás egység (U), meghívási konnektor (IU)
- kompozíciós konnektor
  - vezérlési struktúra (szekvencia, elágazás, ciklus)
- kompozit komponens

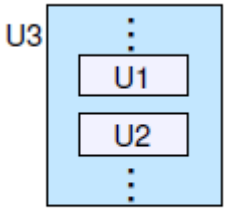


Kompozíciós mechanizmusok

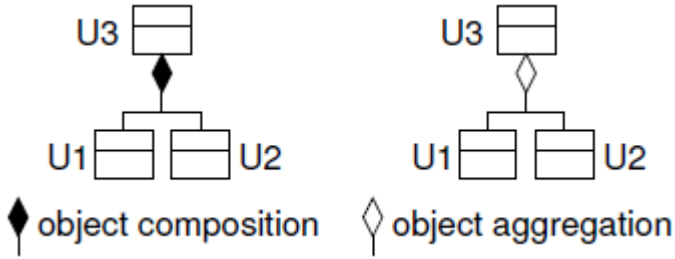
1. tartalmazás
2. kiterjesztés
3. kapcsolódás
4. koordináció

Tartalmazás

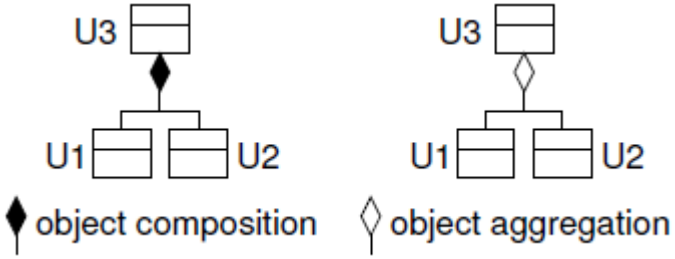
- viselkedési egységek a kompozit viselkedési egység definíciójába kerülnek
- a kompozit egység viselkedését az összetevő egységek viselkedésének felhasználásával definiálják, de a pontos definíció mechanizmusonként változhat
- pl. egymásba ágyazott függvények, eljárások, modulok, osztályok, valamint kompozíció és aggregáció
- az UML csak osztályokra értelmezi (viselkedésről van szó)



(a) Generic



Tartalmazás (példa)



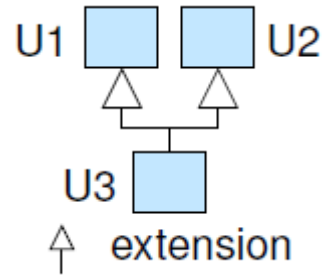
- kompozíció
  - a tartalmazó objektum a tartalmazott objektumok életciklusát kezeli
- aggregáció
  - a tartalmazott objektumok életciklusa független a tartalmazótól

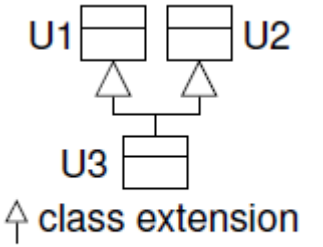
```
class contained
{ ... }
class compose
{ public:
  ...
  private:
    contained first;
    contained second;
};
```

```
class aggregate
{ public:
  ...
  void setContained(contained *,contained *);
  private:
  contained *first;
  contained *second;
};
void aggregate::setContained(contained *c1,contained *c2)
{ first=c1; second=c2;  }
```

Kiterjesztés

- a kompozit viselkedési egységet az összetevő egységek viselkedésének kiterjesztésével definiálják
- például
  - többszörös öröklődés, ami két osztályból egy harmadikat hoz létre
  - aspektus szövés, ami egy osztályból és egy aspektusból egy osztályt hoz létre
  - …
- UML csak osztályokra értelmezi, aspektusokra például nehézkes lenne





(b) UML

Kiterjesztés (példa)

- aspektus szövésnél az aspektus az alapkódban meghatározott programhelyeken (join point) megváltoztatja (advice) a kódot az aspektus leírás (pointcut) szerint

(AspectJ)

```
public class application{ ...
  public void display(){
  System.out.println("Mode");
  } ...
}
```

Output before weaving

```
Mode
```

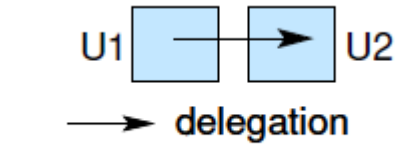
```
public aspect trace{
  pointcut log():
    execution(public * *.display(..));
  before( ) : log( ){ //before advice
    System.out.println("Entering---");}
  after( ) returning : log( ){ //after advice
    System.out.println("Exiting---");}
}
```

Output after weaving

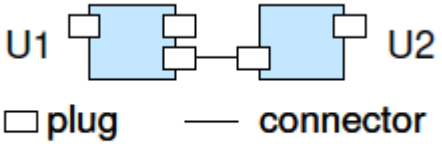
```
Entering---
Mode
Exiting---
```

Kapcsolódás

- a kompozit viselkedést az összetevő viselkedések interakciójával definiálják
- az egységek meghívják egymás viselkedését
  - direkt vagy indirekt
- üzenetküldés, szoros kapcsolódás
- például
  - objektum delegáció
  - architekturális egységek kapcsolása csatolókkal



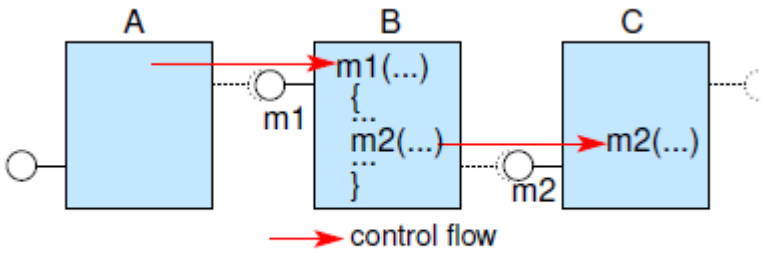
(a) Direct message passing



(b) Indirect message passing

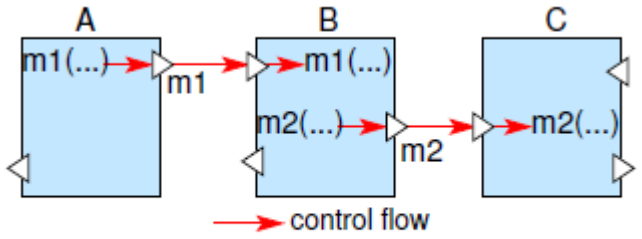
Kapcsolódás

- direkt üzenetátadás
  - objektumok közvetlenül meghívják a metódusokat (delegáció)



(a) Objects

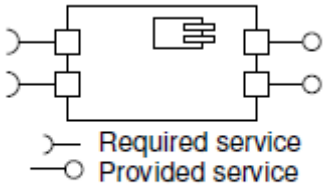
- indirekt üzenetátadás
  - csatolókon keresztül



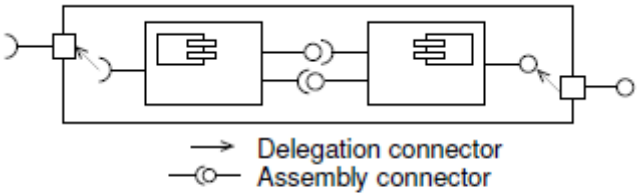
(b) Architectural units

Kapcsolódás

- UML csak komponensekre értelmezi, osztályokra nem
  - elvárt és szolgáltatott interfész
  - kapcsolódás (assembly)
  - port forwarding (delegation)



(a) Component



(b) Composite component

- Mj: UML az osztályokra csak asszociációt értelmez, a metódus hívásokkal összekapcsolt osztályokra nincs jelölése

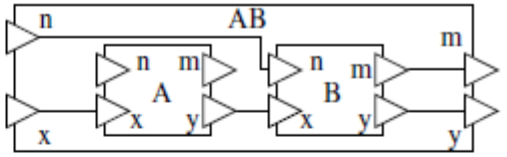
Kapcsolódás (példa)

- architekturális egységek kapcsolása csatolókkal

(ArchJava)

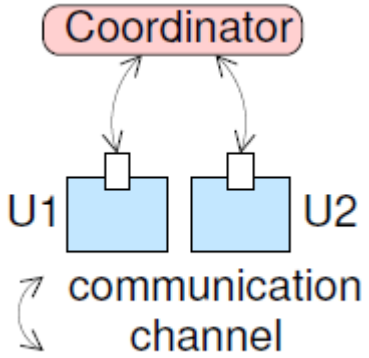
```
component class A{
  port x{ requires int readNum();}
  port y{ provides int add();}
  port n{ requires char readTxt();}
  port m{ provides void printChar();}
  //implementation of provided methods
    ...
}
```

```
component class B{
  port x{ requires int add();}
  port y{ provides int sqr();}
  port n{ requires char readTxt();}
  port m{ provides void printChar();}
  //implementation of provided methods
    ...
}
```




```
component class AB{
  port x{ requires int readNum();}
  port y{ provides int sqr();}
  port n{ requires char readTxt();}
  port m{ provides void printChar();}
  private final A a = new A();
  private final B b = new B();
  connect a.y,b.x;
  glue n to b.n;
  glue x to a.x;
  glue m to b.m;
  glue y to b.y;
}
```

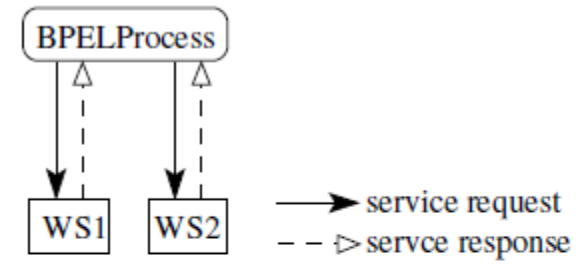
Koordináció



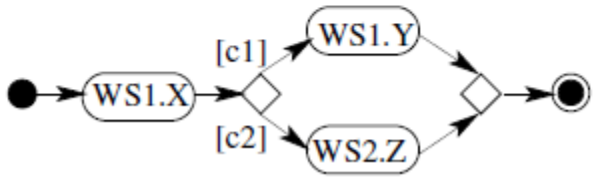
- a kompozit viselkedés az összetevő viselkedések koordinációjával áll elő
- a koordinációt egy koordinátor végzi, adat vagy kontrol csatornákon kommunikál az összetevőkkel, az összetevők egymással közvetlenül nem kommunikálnak
- a kapcsolódásos kompozícióval ellentétben a koordinációs kompozíció teljesen szétválasztja az összetevőket
- például: párhuzamos folyamatok adatkoordinációja, web service orchestration
- UML-ben nincs ez a fogalom, így jelölése sincs

Koordináció (példa)

- a web szolgáltatás WSDL nyelven leírt interfészén keresztül szolgáltatásokat nyújt
- a meghívások sorozatát BPEL munkafolyamat leíró nyelven definiálják (orchestration), amit munkafolyamat motor hajt végre



Workflow defined by BPELProcess:




```
<process name="BPELProcess"..>
<!--Participants (WS1 and WS2)-->
<partnerLinks>
 <partnerLink name="WS1"../>
 <partnerLink name="WS2"../>
</partnerLinks>
<!--Request/response vars of participants-->
 <variables> ... </variables>
 <sequence>
  <invoke..partnerLink="WS1"..operation="X"../>
  <if> <condition> <!--c1--> <condition>
   <sequence>
   <invoke..partnerLink="WS1"..operation="Y"../>
   </sequence>
  <else>
  <if> <condition> <!--c2--> <condition>
   <sequence>
   <invoke..partnerLink="WS2"..operation="Z"../>
   </sequence>
  </if> </else> </sequence>
</process>
```

Kompozíciós mechanizmusok

	Unit of Composition	Composition Mechanism			
		Containment	Extension	Connection	Coordination
Programming View	Function	Function nesting		Higher–order function Function call	
	Procedure	Procedure nesting		Procedure call	
	Class	Class nesting Object composition Object aggregation	Multiple inheritance	Object delegation	
	Mixin		Mixin inheritance		
	Mixin/Class		Mixin–class inheritance		
	Trait		Trait composition	Trait composition	
	Trait/Class		Trait–class composition	Trait–class composition	
	Subject		Subject composition		
	Feature		Feature composition		
	Aspect/Class		Weaving		
	Module	Module nesting		Module connection	
CBD View	Architectural unit			Port connection	
	Fragment box		Invasive composition	Invasive composition	
	Process			Channels	Data coordination
	Web service				Orchestration (Control coordination)
	Encapsulated component				Exogenous composition (Control coordination)

Construction View

Algebrai kompozíciós mechanizmusok

- szisztematikus kompozícióra van szükség, ehhez definiálják az algebrai kompozíciós mechanizmust

Egy kompozíciós mechanizmust valamilyen típusú kompozíciós egységre lehet alkalmazni, és eredményül valamilyen kompozíciós egységet kapunk. Algebrai kompozíciós mechanizmus esetén a kompozíciós egységek mindegyike azonos típusú.

- ez jó, mert hierarchia alakítható ki (szisztematikus)
- konstrukciós nézetben az ilyen mechanizmus kívánatos, mert komponens algebrát alkot

Algebrai kompozíciós mechanizmusok

	Composition Mechanism			
	Containment	Extension	Connection	Coordination
Algebraic	Function nesting Procedure nesting Module nesting Class nesting Object composition Object aggregation	Multiple inheritance Mixin inheritance Trait composition Subject composition Feature composition Invasive composition	Higher–order function Trait composition Port connection Invasive composition Channels	Exogenous composition
Non–Algebraic		Mixin–class inheritance Trait–class composition Weaving	Function call Procedure call Module connection Object delegation Trait–class composition	Data coordination Orchestration

- csak az exogén koordináció algebrai
- adat koordináció nem, mert csak folyamat halmazt eredményez
- orchestration sem, mert munkafolyamatot eredményez (ezért néhány BPEL szerkesztő kikényszeríti a web szolgáltatásba csomagolást)

Kompozíciós operátorok

- automatizálható kompozícióra van szükség, ehhez definiálják a kompozíciós operátorokat
- hasonlóan a függvényekhez $f:X \rightarrow Y$ és $g:Y \rightarrow Z$ kompozíciója $h:X \rightarrow Z$, $h(x)=g(f(x))$ és ekkor h alkalmazható minden $X \rightarrow Y$ és $Y \rightarrow Z$ függvényre
- ha vannak kompozíciós operátorok, akkor nincs szükség „ragasztó” kódra, a hierarchikus kompozíció automatikusan működhet

Kompozíciós operátorok

Composition Operator?	Algebraic Composition Mechanism			
	Containment	Extension	Connection	Coordination
No	Function nesting Procedure nesting Module nesting Class nesting Object composition Object aggregation	Multiple inheritance Trait composition Feature composition Invasive composition	Trait composition Port connection Invasive composition Channels	
Yes		Mixin inheritance Subject composition	Higher–order function	Exogenous composition

- tartalmazásnál nincs, mert tetszőlegesen kombinálhatók
- kiterjesztésnél nincs, ha a konfliktust fel kell oldani
- kapcsolódásnál nincs, ha ad hoc módon összekapcsolhatók a komponensek

Kompozíciós operátorok

- például a mixin öröklődésnél definiálható operátor, amit rekord kombináció bináris operátoraként tekinthetünk

- (MixedJava)

```
mixin A{
 m1(){...} m2(){...}
 m5(){//print Alpha}
}
```

```
mixin B{
 m3(){...} m4(){...}
 m5(){//print Beta}
}
```

```
//two composition expressions
mixin AB=A compose B;
mixin BA=B compose A;
```

```
mixin AB{
 m1(){...} m2(){...}
 m3(){...} m4(){...}
 m5(){//print Alpha}}
```

```
mixin BA{
 m1(){...} m2(){...}
 m3(){...} m4(){...}
 m5(){//print Beta}}
```

Összefoglalás

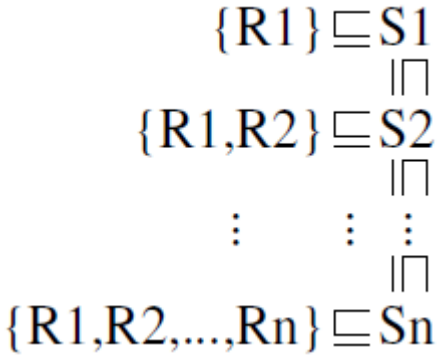
- kompozíciós mechanizmusok kategorizálása
  - tetszőleges viselkedési kompozíciós egységekre
  - szisztematikus és automatizálható kompozíciókészítéshez
- kívánatos, hogy
  - a kompozíciós mechanizmus exogén legyen (a komponensen kívül legyen)
  - a kompozíciós mechanizmusnak külön mechanizmusai legyenek az adatfolyamra és vezérlésfolyamra
  - a kompozíciós nyelv nyújtson lehetőséget magasabb szintű absztrakciók létrehozására

Inkrementális kompozíció követelményspecifikációból

forrás:

- Constructing Component-based Systems Directly from Requirements using Incremental Composition. K.-K. Lau, A. Nordin, T. Rana and F. Taweel. In *Proceedings of Thirty-sixth Euromicro Conference on Software Engineering and Advanced Applications*, pages 85-93, IEEE, 2010.

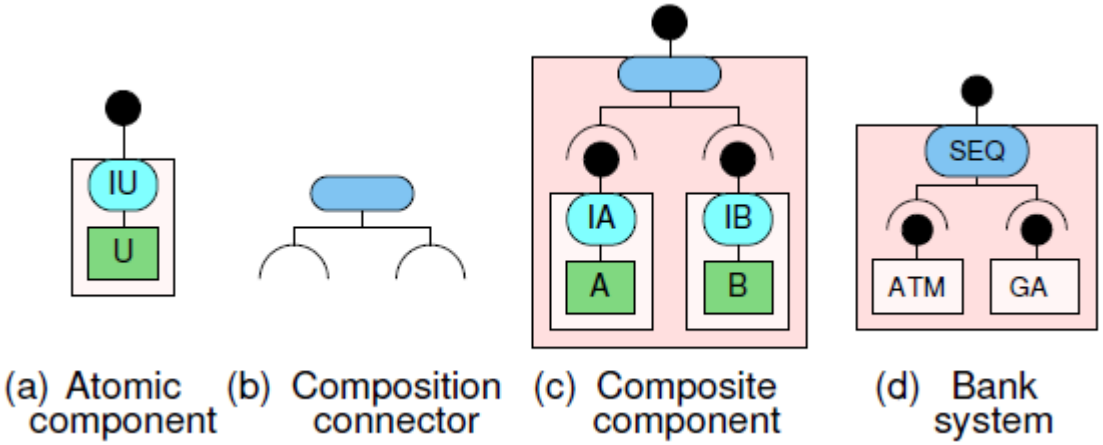
Inkrementális kompozíció



- egy módszer a komponensek beazonosítására és a kompozíció létrehozására
- azzal a feltételezéssel, hogy
  - a nyers követelményspecifikáció részei (Ri) leképezhetőek részleges komponens architektúrákra (Si), ami komponenseket és kompozíciós struktúrákat tartalmaz
  - a részleges komponens architektúra újabb követelmény figyelembe vételével kibővíthető úgy, hogy a kibővített architektúra kielégíti az új követelményt és továbbra is kielégíti a korábban feldolgozott követelményeket

Inkrementális kompozíció

- alkalmazott kompozíciós mechanizmus: exogén koordináció

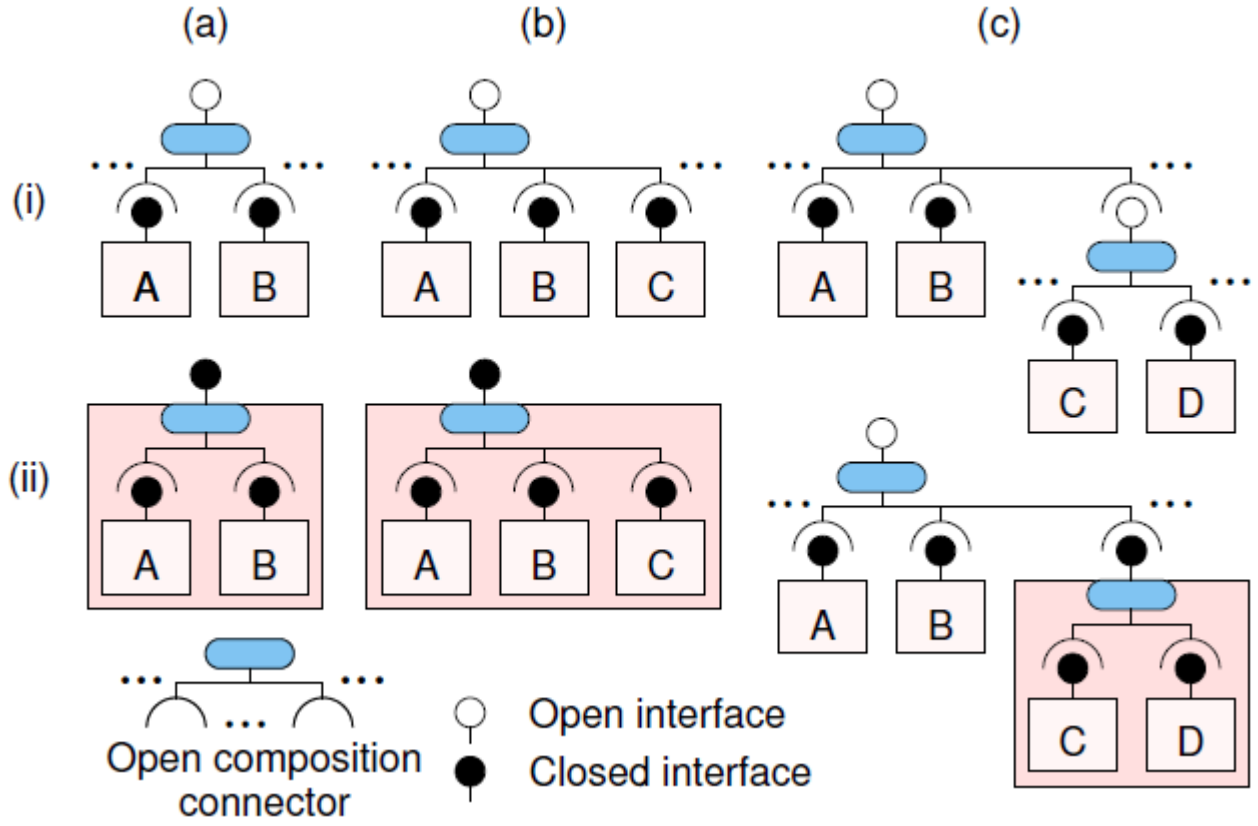


- kompozíciós konnektorok
  - SEQ: szekvencia
  - PIPE: mint szekvencia, csak adatot is átad
  - SEL: szelektor elágazás
  - LOOP: ciklus
  - GUARD: vezérlés feltételes letiltása

Inkrementális kompozíció

- inkrementális tervezés támogatása
  - kompozíciós konnektorok számossága nyitott lehet (jele …)
  - kompozíció interfésze nyitott vagy zárt lehet
- a nyitott kompozíciós konnektor
  - nyitott kompozíciót és nyitott interfészt eredményez
- a zárt kompozíciós konnektor
  - zárt kompozíciót és zárt interfészt eredményez
- nyitott kompozíciót csak akkor lehet lezárni, ha minden komponense zárt

Inkrementális kompozíció



- ez a megközelítés biztosítja, hogy a komponens architektúra inkrementális lehessen
  - befoglalás miatt új komponens nem változtatja a régit
  - hierarchia miatt a régi követelmények is teljesülnek

Információ kinyerése követelményekből

- szokásos technika a követelmény elemeinek hozzárendelése a tervezési modell elemeihez
  - például objektum orientált tervezésben a főnevek objektumokhoz vagy osztályokhoz köthetők, míg az igék objektumok közötti üzenetekhez
- az itt alkalmazott kompozíciós modell elemei
  - a számítás: transzformáció vagy függvény
  - vezérlési folyam: számítások közötti végrehajtás áramlás (viselkedés részlet)

Információ kinyerése követelményekből

- információ kinyerés igékből

Category of verbs	Denotes	Examples
Computation	Computation (data transformation)	withdraw,deposite, cooking
State	Internal state of components (attribute values of components)	keep,remain
Event	Events (that can trigger computation)	press,cancel,push

Információ kinyerése követelményekből

- információ kinyerés főnevekből

Category of nouns	Denotes	Examples
Conceptual component	Conceptual hooks for components	power tube, authentication
Data	Value or set of values	1,c,integer
State	Attribute name and state	closed,open
Computation	Computation (data transformation)	registration,tranmission, movement

Információ kinyerése követelményekből

- információ kinyerés kifejezésekből

Category of phrases	Denotes	Examples
Descriptive expression	May denote components or computations	"the earlier date" may denote date or compareDate( )
Predicate	Computations - operations that can be true/false	is enabled,is invalid
Control structure	Control structure	if,then,else,while, iterate,loop,after

Minta alkalmazás (ATM) követelményei

R1	The ATM will service one customer at a time. A customer will be required to insert an ATM card and enter a personal identification number (PIN).
R2	A customer must be able to make a cash withdrawal from the linked account. Approval must be obtained from the bank before cash is dispensed.
R3	A customer must be able to deposit cash to the linked account that can be inserted to the cash slot. Approval must be obtained from the bank before physically accepting the cash.
R4	A customer must be able to make a transfer of money between any two accounts originated from the linked account.
R5	A customer must be able to make a balance enquiry of the linked account.
R6	If the customer fails to be authenticated, the card will be rejected.
R7	After each transaction, the ATM will display and print a receipt containing the transaction information.

Inkrementális kompozíció

R1 A customer will be required to insert an ATM card and enter a personal identification number (PIN).

1. lépés:
számítások azonosítása és komponensek választása

- azonosított: „insert card”, „enter PIN”
- létező vagy új komponens keresése
  - több számítás összevonása egy komponensbe vagy
  - egy számítás szétvágása több komponensre
- döntés
  - CardReader és PinReader komponensek

Inkrementális kompozíció

R1 A customer will be required to insert an ATM card and enter a personal identification number (PIN).

2. lépés:
vezérlési folyamok azonosítása és kompozíciós konnektor választása (ha több komponensünk van)

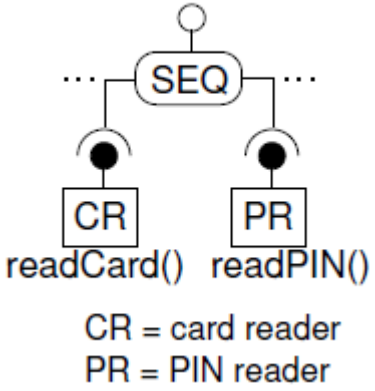
- azonosított: „and”
- kompozíciós konnektor választása
  - szekvencia, vagy pipe ha adattovábbítás is kell
  - szelektor elágazáshoz
  - guard vagy loop az adaptáláshoz
  - konnektorok kombinációja is lehetséges
- döntés
  - szekvencia

Inkrementális kompozíció

> **R1** A customer will be required to insert an ATM card and enter a personal identification number (PIN).

3. lépés:
részleges architektúra létrehozása

- az előző két lépés alapján megkonstruálható



Inkrementális kompozíció

R1 A customer will be required to insert an ATM card and enter a personal identification number (PIN).

4. lépés:
a részleges architektúra kompozíciója a már korábban létrehozott architektúrával

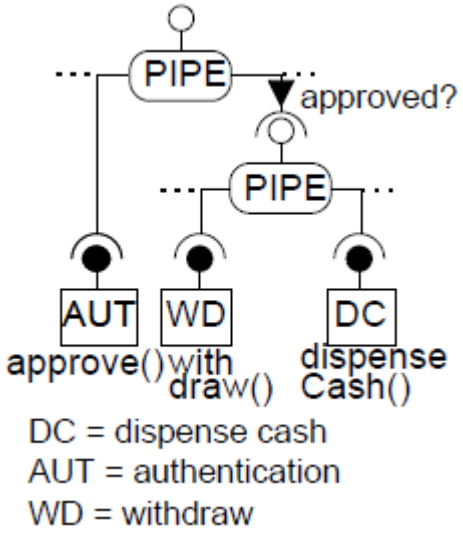
- R1 esetén a korábbi architektúra üres,
- ezért a 4. lépést csak R2 esetén tudjuk demonstrálni

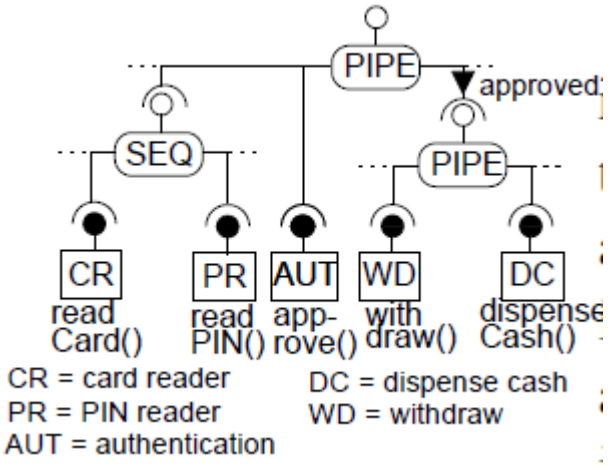
Inkrementális kompozíció

> **R2** A customer must be able to make a cash withdrawal from the linked account. Approval must be obtained from the bank before cash is dispensed.

4. lépés:

- R2 esetén a 2.és 3.lépésben létrehozott részleges és az egyesített architektúra:





Inkrementális kompozíció

4. lépés:

- a kompozíció nem mindig lehetséges, mert nincs megfelelő kompozíciós pont
- ilyenkor elhalasztjuk későbbre, amikor már lehet
- ha sohasem lehet, akkor valószínűleg a követelmény specifikációval van a baj és konzultálni kell a megrendelővel

Inkrementális kompozíció

4. lépés:

- minden inkrement esetén több kompozíciós konnektor több kompozíciós pontja közül lehet választani
- a választás a követelmény specifikáció alapján történik

Inkrementális kompozíció

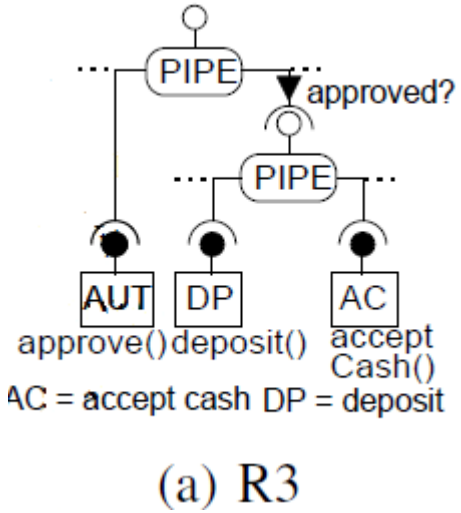
5. lépés:
a rendszer teljes architektúrájának véglegesítése

- az összes követelmény feldolgozása után
- még mindig lehetnek nyitott kompozíciós pontok, így lehet finomítani, adaptálni, optimalizálni
  - komponensek összevonása, konnektorok összevonása, konnektorokhoz adaptációk hozzáfűzése a követelményekből, …
- végül a nyitott kompozíciós pontok lezárása

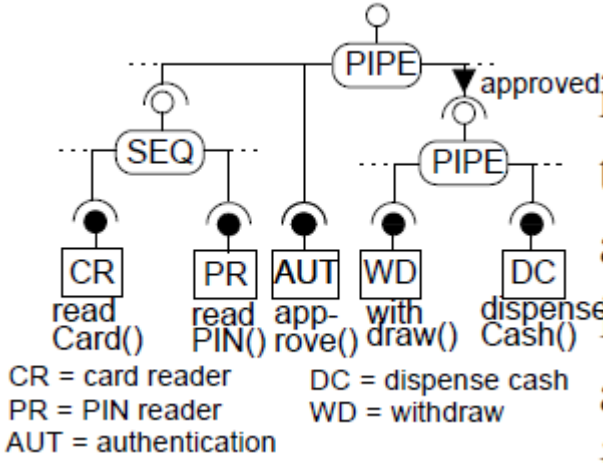
Inkrementális kompozíció

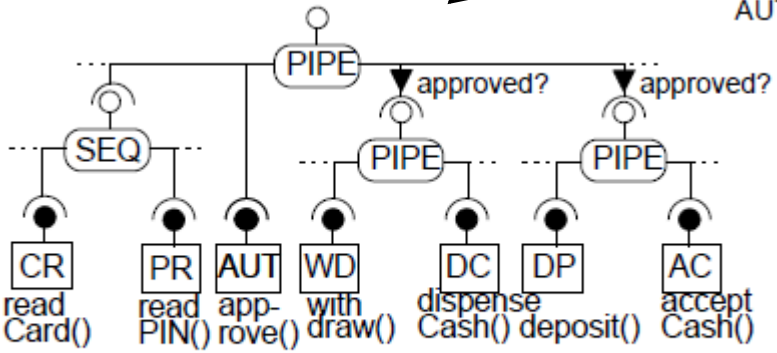
R3	A customer must be able to deposit cash to the linked account that can be inserted to the cash slot. Approval must be obtained from the bank before physically accepting the cash.

R3 és az utána egyesített architektú
(az AUT komponenst nem kell duplikálni)



(a) R3

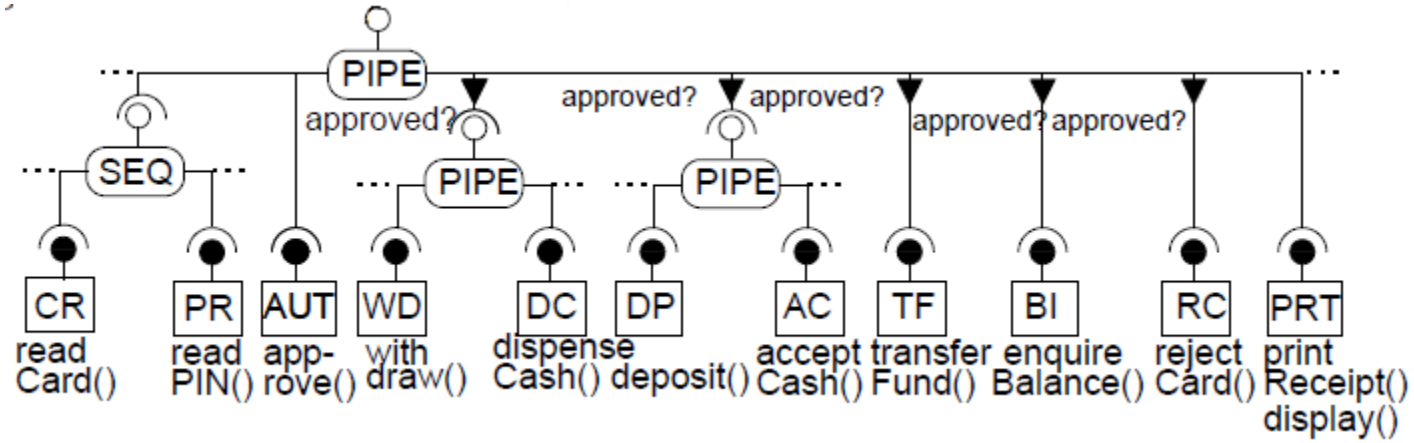




(b) {R1,R2,R3}

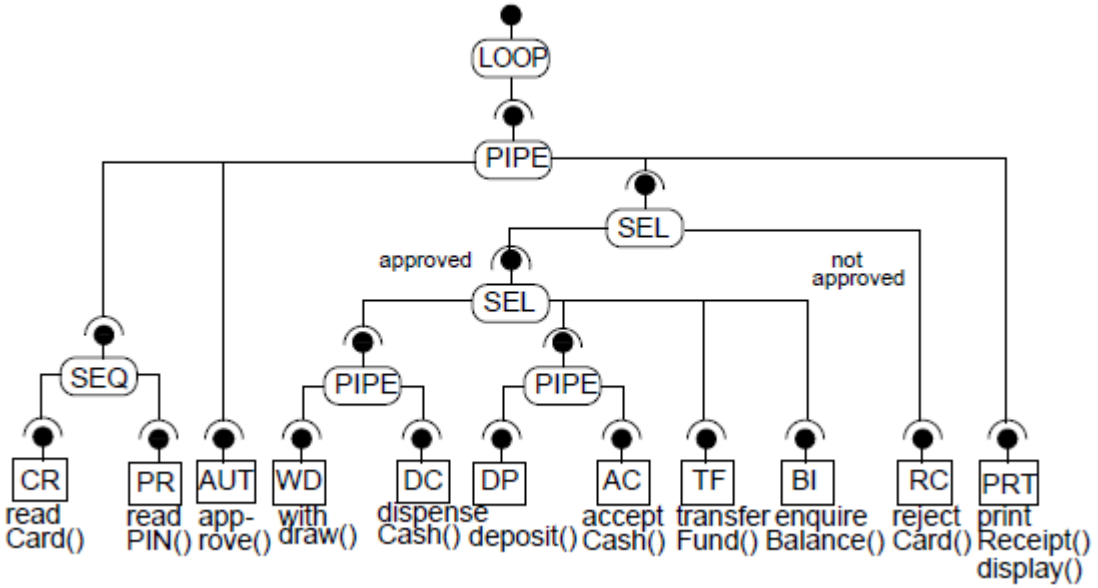
Inkrementális kompozíció

- az összes követelmény feldolgozása után kapjuk



Inkrementális kompozíció

- optimalizálás
  - approved? guard-ok összevonása egy szelektorba
  - loop hozzáadása a legfelső szintre, hogy újra üzemképes legyen (nem volt benne a specifikációban)
- ezt kapjuk:



Összefoglalás

- egy megközelítés a kompozíció elkészítésére
- inkrementális
- heurisztikát és intuíciót tartalmaz
- de a lépéseket és szabályokat megfogalmazta
- nehézsége
  - ha nem találni kompozíciós pontot
  - ha több kompozíciós pont közül kell választani
  - természetes nyelvi elemek azonosítása