

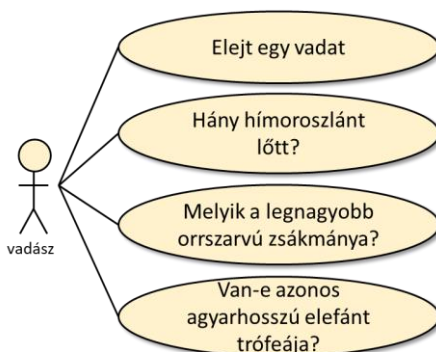
## 9. Modellezés I.

1. Egy vadász (név és az életkor) számos trófeát gyűjtött. Egy trófea az elejtett vad néhány adatát mutatja meg: a vad fajtáját (elefánt, orrszarvú, oroszlán), az elejtés helyét és dátumát, az elejtett vad tömegét (kg-ban), és valamilyen különleges adatot (elefántok esetén agyarainak hosszát külön-külön (cm-ben), orrszarvúaknál a szarvának tömegét, oroszlánoknál a vad nemét).

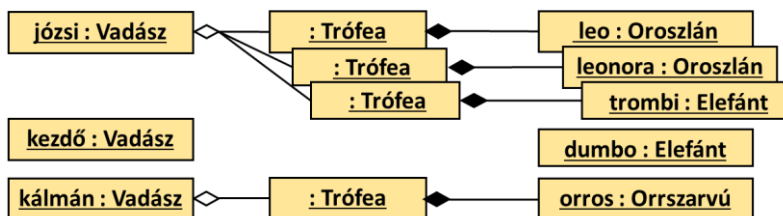
Adjunk választ az alábbi kérdésekre:

- Hány hímoroszlánt lőtt egy adott vadász?
- Melyik egy vadász legnagyobb szarv/testtömeg arányú orrszarvú zsákmánya?
- Van-e egy vadásznak azonos agyarhosszú elefánt trófeája?

Készítsünk először egy használati eset diagrammot.



Vázoljunk fel egy lehetséges objektum diagramot.

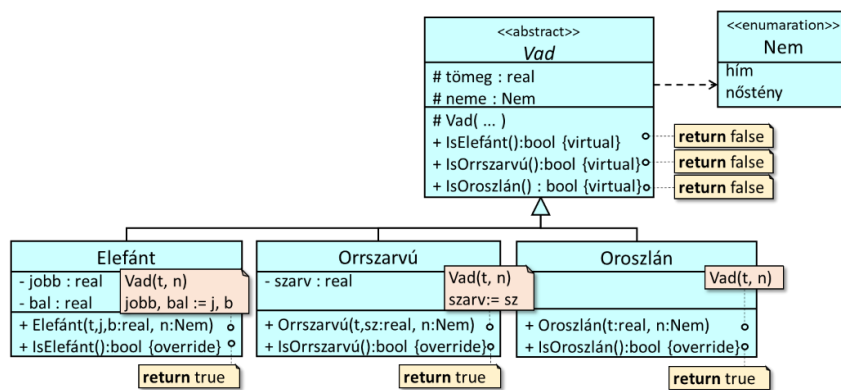


Egy vadász több trófeát is birtokolhat, és minden trófeához tartozik egy vad, de vannak olyan vadak, amelyek nem részei egyetlen trófeának sem. Jelenleg három féle vaddal számolunk. A tömeg, és a vad nemének megadása általános, mindenféle vadra érvényes adatok, ezért az űsosztályban jelennek meg; az agyarak viszont az elefántok specifikus adata, a szarv pedig az orrszarvúaké.

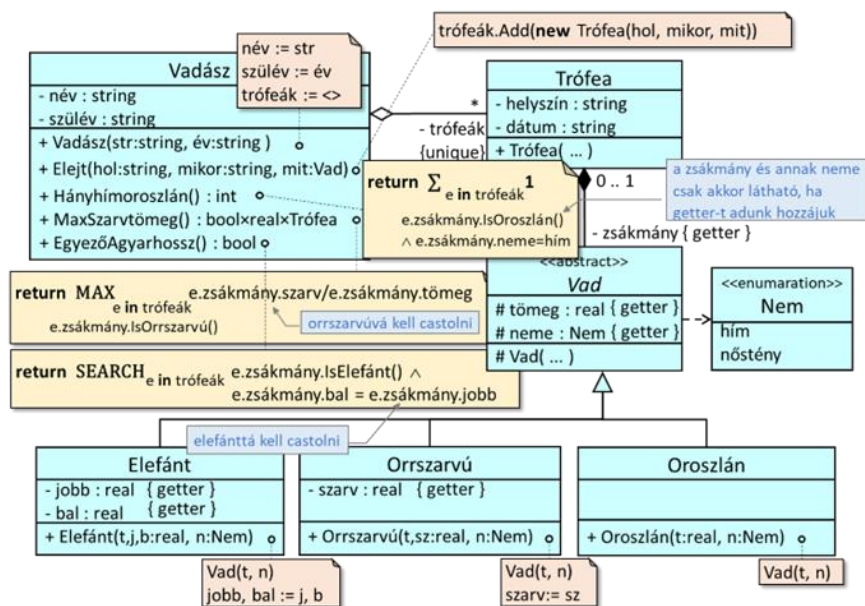
Alkalmazunk az osztálydiagram ábrázolásánál néhány egyszerűsítést.

Egy konstruktor paraméterlistájában a (...) jelölés arra utal majd, hogy a létrehozandó objektum összes adattagjának egy paraméter ad kezdőértéket, így az ilyen konstruktorokat a modellben nem kell részletesen definiálni. Például a `Vad( ... )` jelöli a `Vad( tömeg:real, n:Nem ) { this.tömeg:=tömeg; this.neme := n }` definíciót.

Származtatás esetén feltételezhetjük, hogy rendelkezünk olyan `IsAlosztály():bool` metódusokkal (C#-ban az `is` operátorral), amelyek el tudják dönteni, hogy egy objektum egy adott alosztály példánya-e, még akkor is, ha az objektum hivatkozását az űsosztály típusú változó őrzi. Ezeket a modellben nem fogjuk explicit módon feltüntetni.



Egy trófeának része az a vad (kompozíció), amit a vadász elejtett. A Trófea és Vad objektumpár közti kapcsolatot a Trófea konstruktora hozza létre úgy, hogy paraméterként megkapja meg az elejtett vadat (a zsákmányt) az elejtés helyszínével és dátumával együtt. A vadász birtokolja a trófeáit (aggregáció), egy trófeához egy vad tartozik (kompozíció). A Vadász konstruktora a név és kor adatok kitöltéséért felel. Egy Vadász objektum és egy Trófea objektum közti kapcsolatot az Elejt(mit:Vad, hol:string, mikor:string) metódus hozza létre úgy, hogy ez példányosítja a Trófea típusú objektumot (helyszín, dátum, vad), és helyezi el annak hivatkozását a Vadász objektum trófeák nevű gyűjteménybe.



A feladat kérdései mind egy adott vadászra vonatkoznak, ezért az azokra válaszoló metódusokat a Vadász osztályban vezetjük be. Mindhárom kérdést olyan algoritmus mintára vezethetjük vissza, ahol az adott vadász trófeáit kell felsorolni. Az első kérdést egy számlálás, a másodikat egy feltételes maximum keresés, a harmadikat egy lineáris keresés válaszolja meg.

Mindhárom esetben megfigyelhető, hogy amikor a trófák felsorolása során az e.zsákmány-ra hivatkozunk, akkor ennek statikus típusa a Vad, de a dinamikus típusa a Vad valamelyik alosztálya. Az IsAltípus() formájú (pl e.zsákmány.IsOroszlán()) metódushívásoknál a dinamikus altípusos polimorfizmus segít a megfelelő IsAltípus() metódus meghívásában (lévén ezek virtuálisok); viszont az e.zsákmány.szarv, e.zsákmány.bal, e.zsákmány.jobb hivatkozásoknál az e.zsákmányt előbb megfelelő (orrszarvú, illetve elefánt) típusúra kell alakítani. Ehhez a C# több nyelvi eszközt is biztosít.

A modell jelzi, hogy mely adatokat kell mindenképpen getter-rel ellátni, de valójában az összes adatához lehetne getter-t biztosítani.

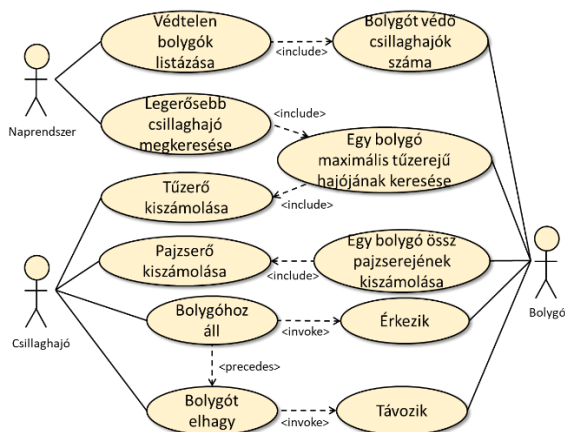
2. A bukott légiók útban vannak a Birodalom központi naprendszere felé Horus Lupercal vezetésével, hogy megdöntsék a Császár hatalmát. Ahogy az áruló sereg közeleg, a birodalomhoz hűséges bolygók egymás után borulnak sötétségbe a naprendszer körül. Az utolsó reményt azok a hűséges légiók jelentik, amelyek a naprendszer bolygóinál gyülekeznek.

A naprendszer bolygóit egyedi nevű csillaghajók védik. Három fajta csillaghajó van: faltörő, partraszálló, és lézerező. Egy csillaghajónak harci képességét a páncélozottsága (egész szám), a pajzserőssége (egész szám), és rajta szolgálatot teljesítő űrgárdisták száma határozza meg.

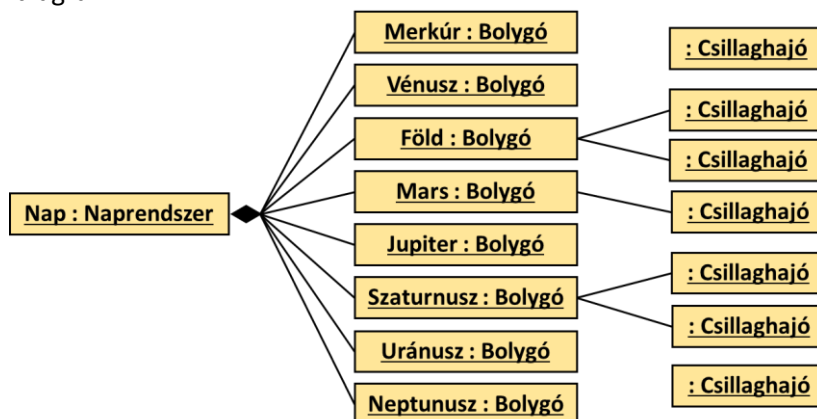
Egy csillaghajó tűzerejét a fajtája és a harci képessége határozza meg:

- A faltörő tűzereje a páncélozottságának a fele.
  - A partraszállóból az űrgárdisták torpedókból élve jutnak át az ellenséges csillaghajók fedélzetére, így a partraszálló tűzereje az űrgárdisták száma.
  - A lézerező a pajzsokról irányítja át az energiát a lézerfegyverekbe, a tűzereje így megegyezik a pajzserejével.
- Tudomásunkra jutott, hogy vannak bolygók, amityeket védtelenül hagyunk, ahol nem állomásozik csillaghajó. Listázzuk ki ezeket a bolygókat!
- Indítunk egy megelőző csapást, amit a legnagyobb tűzerejű csillaghajó vezet majd. Keressük meg ezt a csillaghajót a naprendszerben!
- Szeretnénk megtudni, hogy egy adott bolygónál állomásozó csillaghajóknak mennyi az összes pajzsereje!

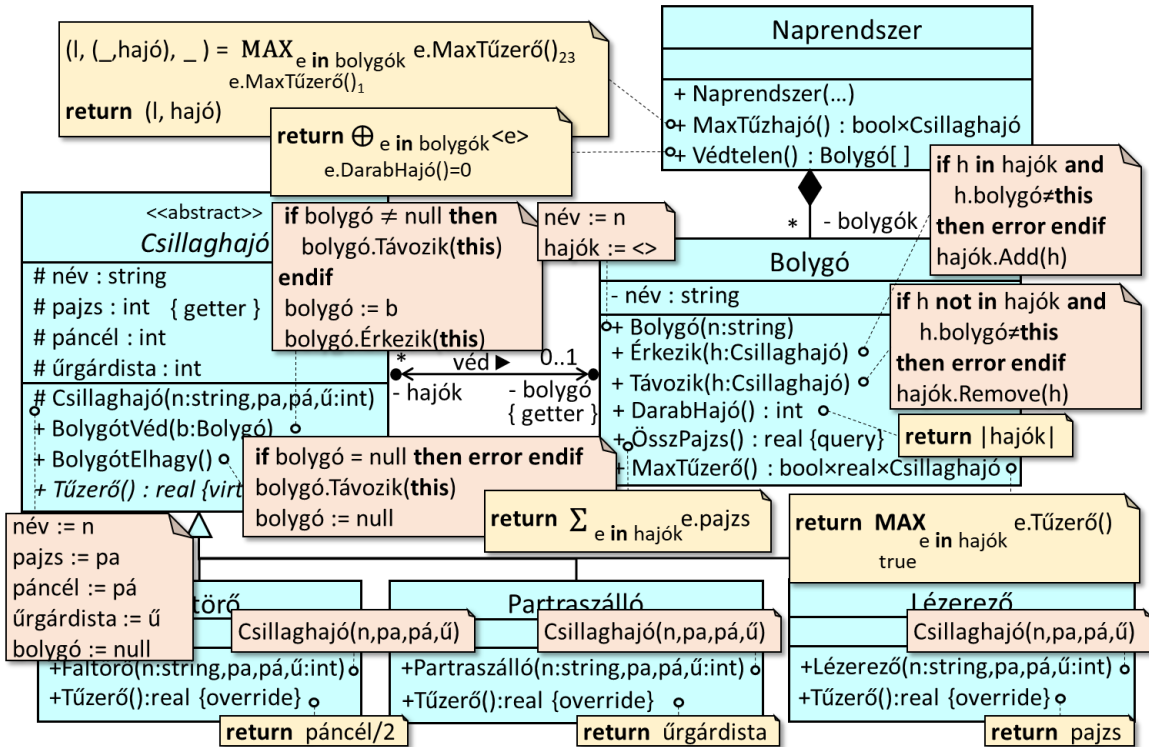
Használati eset diagram



Objektum diagram:



Egy csillaghajó a BolygóhozÁll() metódussal tud egy bolygó védelmére kellni, és a BolygótElhagy() metódussal azt otthagyni. Mivel a csillaghajó közvetlenül nem férhet hozzá a bolygót védő hajók gyűjteményéhez, ezért azt megváltoztatni csak közvetve tudja a Bolygó osztály Véd() és Elhagy() metódusain keresztül, amelyek így ellenőrzött lehetőséget biztosítanak egy bolygót védő hajók gyűjteményének módosítására.



A védtelen bolygókat a Naprendszer Védtelen() metódusában elhelyezett kiválogatás (feltételes összegzés) adja meg. Ehhez felhasználjuk a Bolygó DarabHajó() metódusát, ami egy bolygót védő hajók darabszámát adja meg.

A naprendszer legnagyobb tűzerejű csillaghajóját a Naprendszer osztály MaxTűzhajó() metódusa számolja ki. Ez a feltételes maximumkeresés bolygónként hasonlítja össze az azokat védő legnagyobb tűzerejű hajókat, kihagyva a védtelen bolygókat. Egy bolygótól lekérjük, hogy védi-e hajó, és melyik ezek között legerősebb tűzerejű, és mekkora ez a tűzerő (MaxTűzerő()). A felt.max.ker. feltételét ennek első komponense adja (e.MaxTűzerő()<sub>1</sub>), második és harmadik komponense az összehasonlítandó tűzerő-hajó értékpárokat (e.MaxTűzerő()<sub>23</sub>). Ezen párok összehasonlításnál csak a tűzerőt vesszük figyelembe, viszont a keresés második eredménye (ezt szoktuk a max változóval jelölni) így egy értékpár lesz: a naprendszer legerősebb hajójának tűzerejét, de magát a hajót is tartalmazza. A keresés első eredménye (l) mutatja, hogy van-e hajó a naprendszerben; a harmadik eredményre, ami az a bolygó, amit a legerősebb hajó véd, nincs szükségünk. A Bolygó felhasznált MaxTűzerő() metódusa tehát egy értékhármaszt ad vissza: van-e hajó a bolygónál, mi azok legerősebbikének tűzereje, és melyik ez a hajóé. Ehhez egy azonosan igaz feltételű feltételes maximumkeresést (így kezeljük le a védtelen bolygók esetét) használunk, amely egy-egy hajó Tűzerő() metódusát hívja meg, amit viszont a Csillaghajó alosztályai implementálnak.

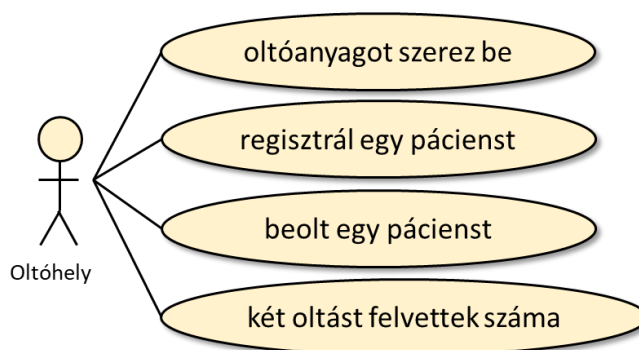
A harmadik kérdést a Bolygó ÖsszPajzs() metódusa válaszolja meg, amely egy olyan összegzés, amihez le kell tudnunk kérdezni egy hajónak a pajzs erejét, amit ezért publikus getter-rel látunk el.

3. Ha egy állampolgár oltakozni akar a COVID-19 ellen, akkor először regisztráltatnia kell magát egy oltóhelyen, majd ott egy adott időpontban felvenni az oltást, feltéve, hogy a páciens által választott oltóanyag rendelkezésre áll. A páciens többféle vakcinával többször is oltathatja magát. Az oltóhelynek ismert a helyszíne, továbbá az, hogy melyik fajta oltóanyagból (ez lehet: pfizer, moderna, astrazeneca de később bővíülhet még a lajstrom) hány felhasználható adag van, és egy adott vakcinánál hány nappal később lehet beadni a második oltást. Nyilvántartjuk, hogy a beoltott páciensek mikor milyen vakcinát kaptak.

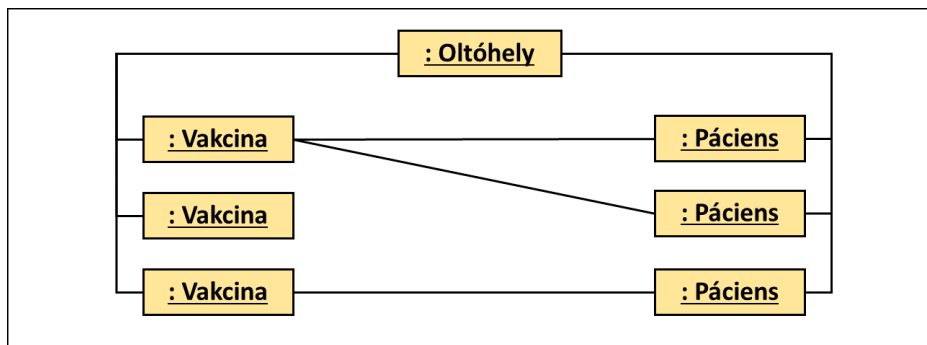
Tegyük lehetővé, hogy

- egy páciens regisztrálhasson az oltóhelyen
- az oltóhely bevételezhessen vakcinát
- rögzíthető legyen az, amikor egy regisztrált páciens egy általa választott vakcinával beoltatja magát
- lekérdezhető legyen, hány olyan páciens van, aki a második oltást is megkapta.

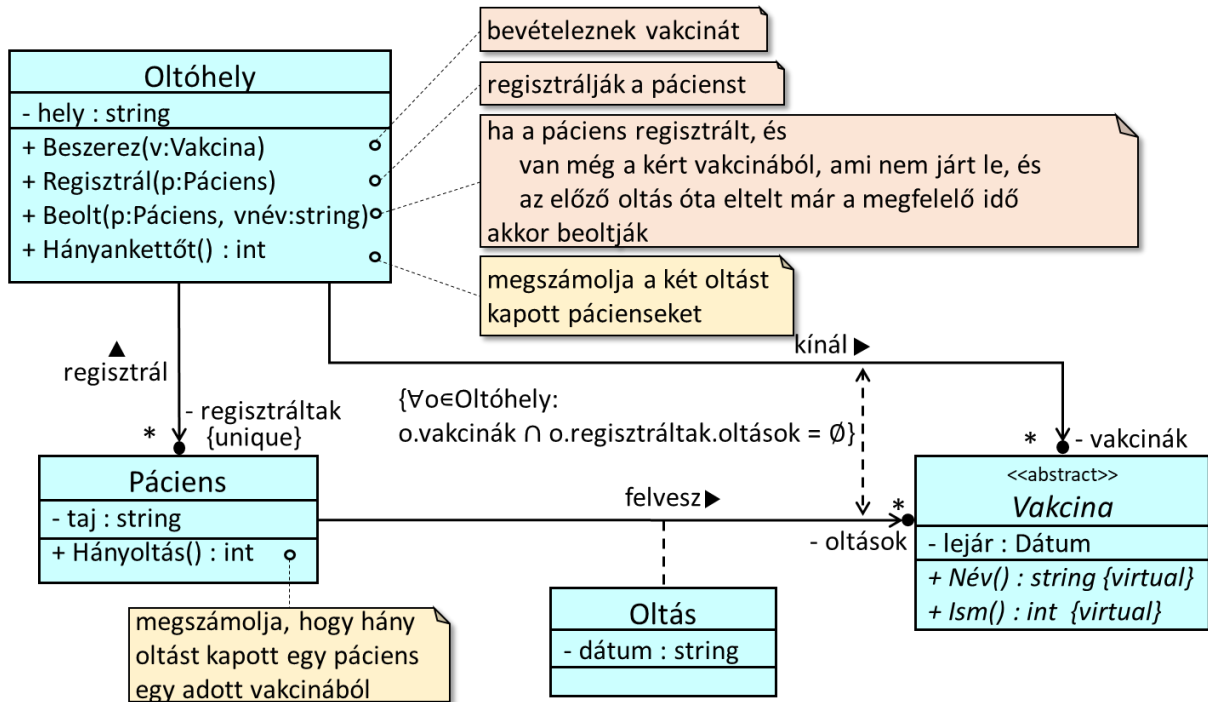
Készítsünk először használati eset diagrammot.



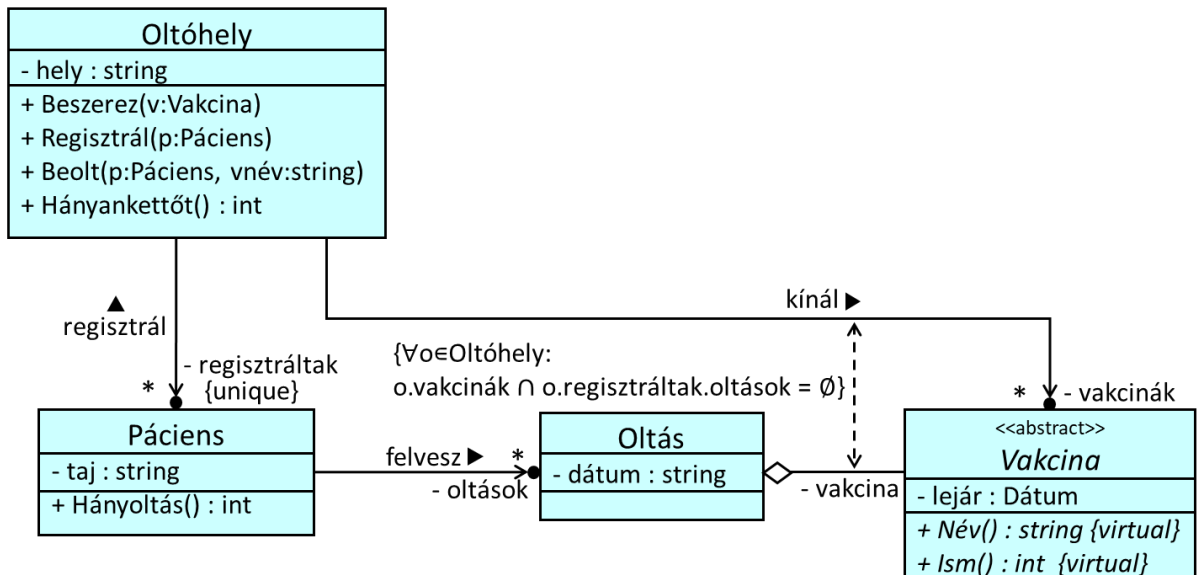
Az objektum diagram pontatlan: egy vakcina nem lehet egyszerre egy oltóhelyen kínálatában is, és az oltóhelyen már felvett vakcina is. Ezt majd az osztálydiagramban jeleznünk kell.



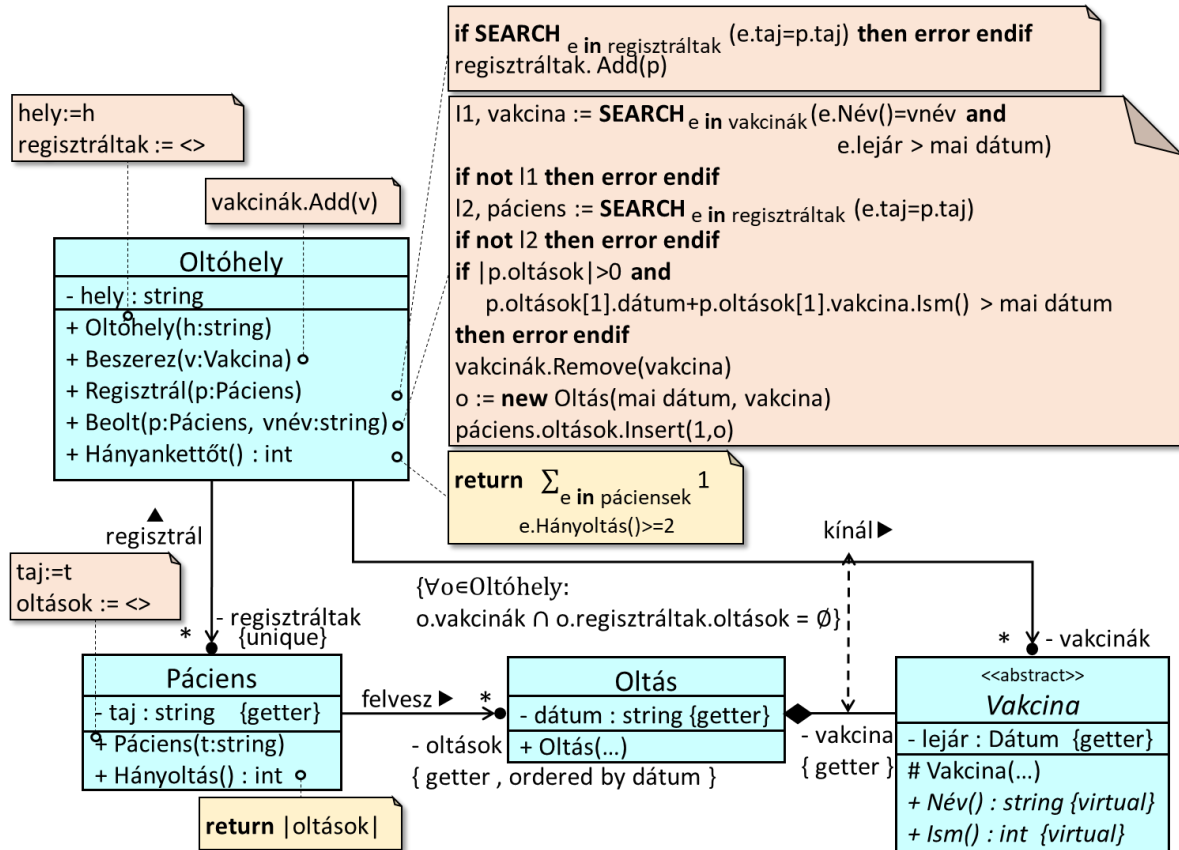
A modell gerincét az Oltóhely, Vakcina, Páciens osztályok kötötti kapcsolatrendszer adja. Az oltások a páciensek és a felvett vakcinák közötti kapcsolatban jelennek meg: ez tehát egy asszociációs osztály.



Elimináljuk az asszociációs osztályt:



Készítsük el a metódusok törzsét:



Az oltóanyag fajtákat származtatással definiáljuk:

