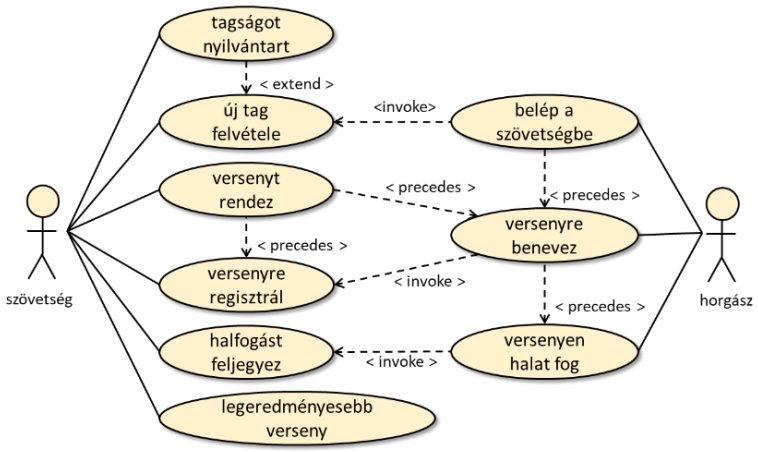
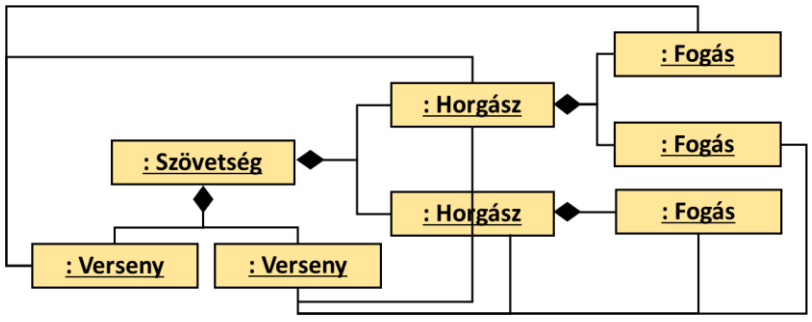
10. Modellezés II.

1. Egy horgászszövetség több horgászversenyt is rendez, amelyekre csak a szövetség tagjai nevezhetnek be; ugyanaz a horgász több versenyen is részt vehet. A versenyeknek ismert a helyszíne, és a kezdő időpontja. A horgászoknak ismerjük a nevét, tudjuk, hogy milyen fogásaik voltak az egyes versenyeken. Egy fogás leírja, hogy melyik versenyen fogták, ki volt a horgász, mi a kifogott hal fajtája és a tömege (kg-ban). A halak fajtája lehet ponty, keszeg, vagy harcsa. A hal értéke a hal tömegének és a halfajta szorzójának (harcsa:3, ponty:2, keszeg:1) szorzata. Melyik a legeredményesebb verseny: ahol mindenki fogott harcsát, és az ilyen versenyek közül itt a legnagyobb a horgászok fogásainak összértéke?

   A szövetség és a horgász használati esetei összekapcsolódnak. Egy horgásznak a szövetségbe történő belépése a szövetség tagságnyilvántartásának része; a szövetség által rendezett versenyre való benevezését a szövetség regisztrálja, de csak a verseny meghirdetése után; a versenyre benevezett horgász a versenyen történt halfogását a szövetség feljegyzi.

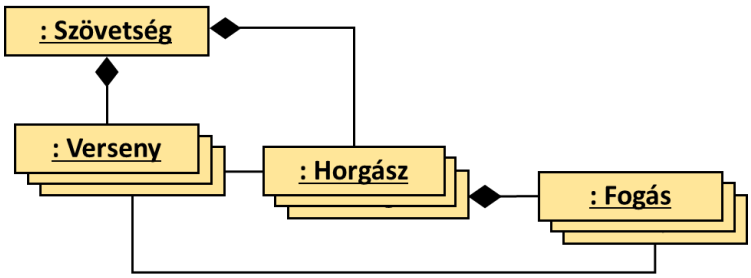


   Első olvasatra négyféle típusú objektummal számolhatunk. Az alábbi objektumdiagramban a szövetségnek két horgász tagja van, és a szövetség két versenyt rendezett. Az első horgász mindkét versenyre regisztrált, és versenyenként egy-egy fogása volt. A második horgász csak a második versenyre regisztrált, ahol egy fogása volt.

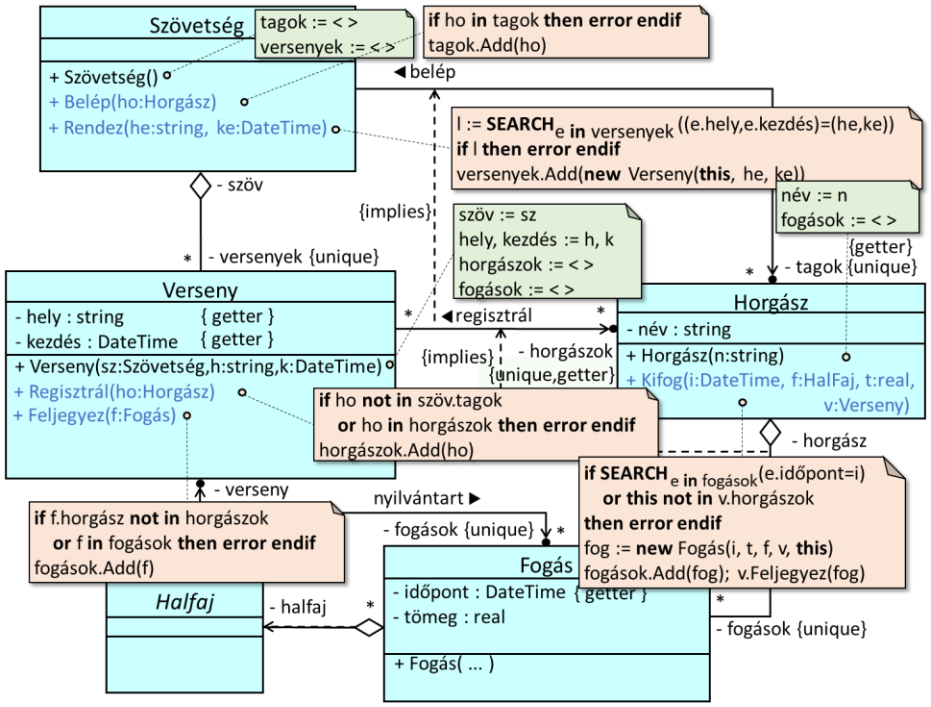


   Talán kifejezőbb az objektumdiagram alábbi (bár nem szabványos) formája, amely több objektumdiagram általánosításaként fogható fel. Ebben jobban látszódnak az egy-sok kapcsolatok, viszont kevésbé képes

megmutatni azt, hogy a horgászok eltérő számú versenyre regisztrálnak, illetve azt, hogy egy horgász több versenyen fog halat, de ugyanazon a versenyen több fogása is lehet.

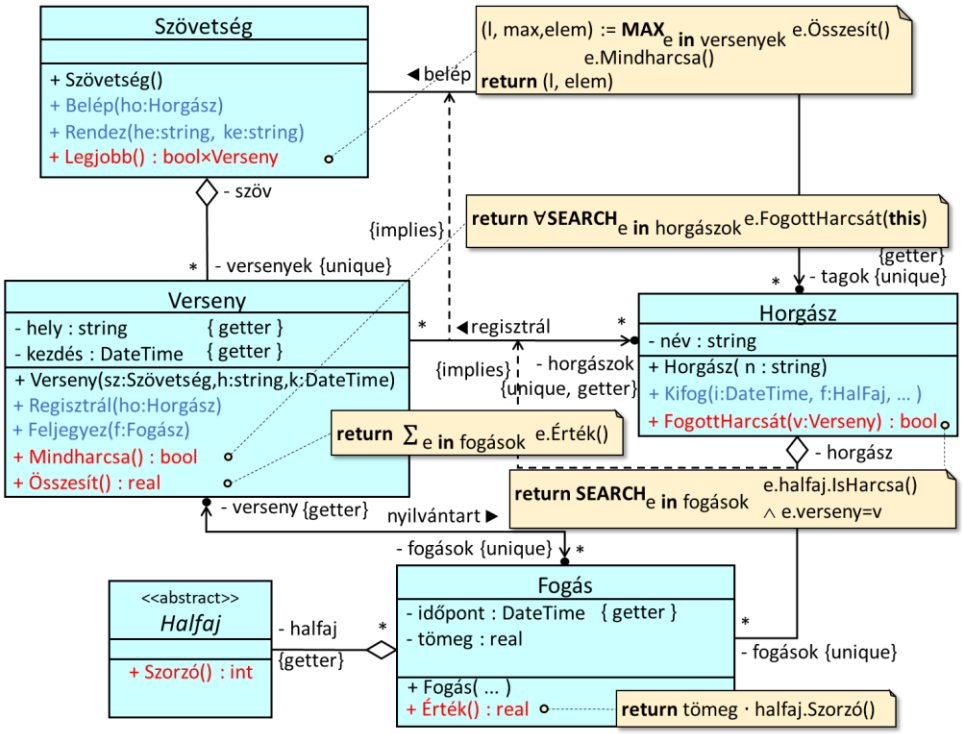


A szövetséghez tartoznak az ahhoz csatlakozó horgászok, és a szövetség által szervezett versenyek. Egy horgász regisztrál egy versenyre, ahol fogásokat ejt. Minden fogáshoz tartozik egy hal, de mivel ezt visszadobják, több különböző fogásnál is szerepelhet. A versenyen nyilvántartják a horgászok fogásait.
Az objektumok között feltételezett hozzátartozási kapcsolatok az osztálydiagramban nem feltétlenül lesznek aggregációval vagy kompozícióval ábrázolva: lehet, hogy „csak” asszociációval. Az asszociációk tervezésével együtt születnek a kapcsolatokat felépítő metódusok is.

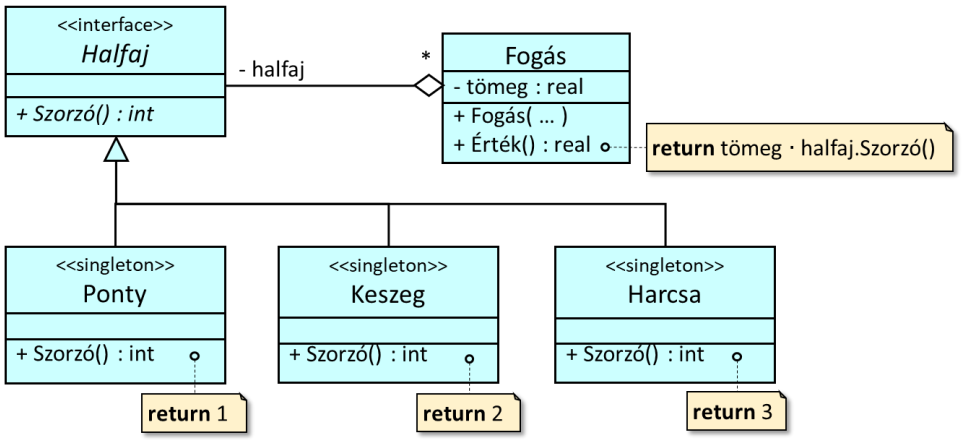


Egy kapcsolat felépítését végző metódusokat a kapcsolat valamelyik osztályában kell elhelyezni. Bizonyos esetekben a kapcsolat felépítésekor kell példányosítani a kapcsolat másik objektumát (pl. rendez, kifog asszociációknál), máskor már létező objektumok között kell kapcsolatot létrehozni (lásd a belép, regisztrál asszociációkat). Ettől függ az, hogy a gyűjtemények „unique” tulajdonságát milyen ellenőrzéssel kell biztosítani: létező objektumok esetén elég az objektum-hivatkozás szintű ellenőrzés (ld. Belép(), Regisztrál()), máskor tartalom szintű ellenőrzés kell (lásd a Rendez(), Kifog()). Ilyenkor kell ellenőrizni az „implies” megszorításokat is.

A feladat kérdésére választ adó Legjobb() metódust a Horgász-szövetség osztályában helyezzük el. Ez a versenyek felsorolására épített feltételes maximumkeresés, amelynek feltételét és értékét a Verseny osztályba telepített metódusok szolgáltatják. Ezek a metódusok a verseny horgászainak felsorolására épülnek: az Összesít() egy összegzés, a Mindharcsa() egy optimista lineáris keresés. Az Összesít() a Horgász osztály Összérték() metódusa által szolgáltatott értékeket adja össze; a Mindharcsa() a Horgász osztály FogottHarcsát() metódusát használja a keresés feltételeként. Az újabb metódusok (Összérték(), FogottHarcsát()) egy horgász fogásait sorolják fel: az egyik kiszámolja az adott versenyen fogott halak értékét (összegzés), a másik azt vizsgálja, hogy fogott-e harcsát a horgász egy adott versenyen (lineáris keresés).

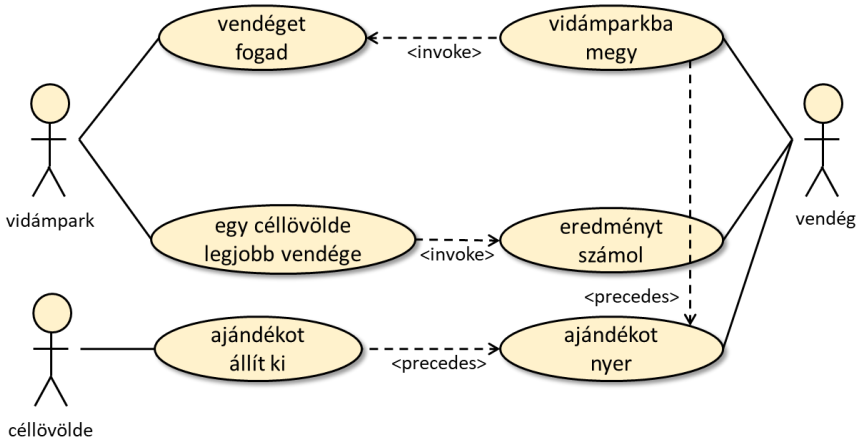


A fogások Érték() metódusa a kifogott hal tömegének és a fajától függő szorzótényezőnek (Szorzó()) szorzatát számolja ki. Annak eldöntéséhez, hogy egy fogás halfaja harcsa-e, az IsHarcsa() metódusra van szükség. A Szorzó() és az IsHarcsa() metódusoknak a hal fajától függő viselkedését a Halfaj alosztályai biztosítják majd. Az így felépülő szerkezet a stratégia tervezési mintának felel meg.

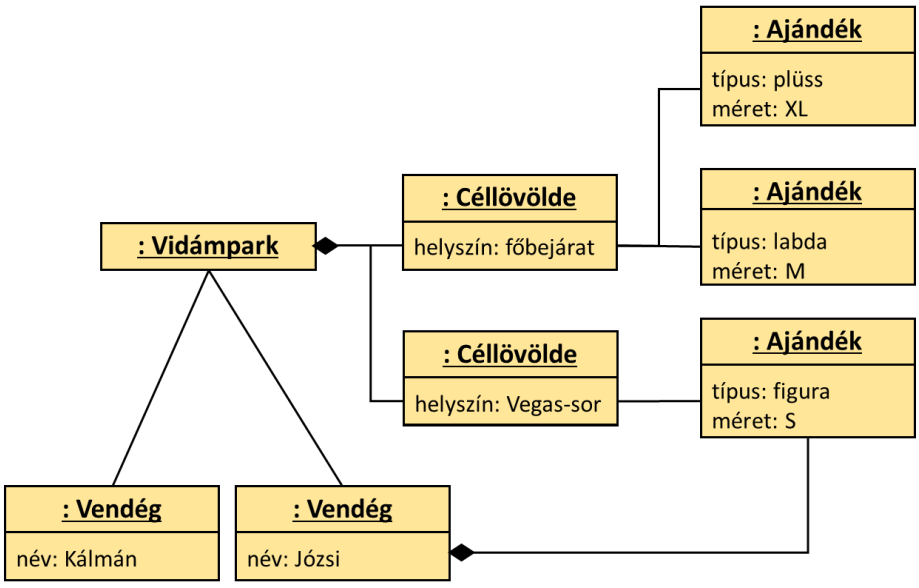


2. Egy vidámparkban a vendégek több céllövöldét is kipróbálhatnak. A céllövöldéknek ismert a helyszínük. Egy céllövöldében egy vendég többször is lőhet, és sikeres találat esetén ajándékot nyer. Egy ajándékról tudjuk, hogy melyik céllövöldében nyerték, mi a típusa (labda, műanyag figura, plüss állat) és mekkora a mérete (S, M, L, XL). Az ajándék értékét úgy számítjuk ki, hogy a típusa után járó pontszámot (plüss állatra 3 pont, műanyag figurára 2 pont, labdára 1 pont) megszorozzuk a mérete után járó szorzóval (az S méret 1 pont, az M 2 pont, az L 3 pont, az XL 4 pont). Nevezzük meg egy céllövölde legjobb céllövőjét!

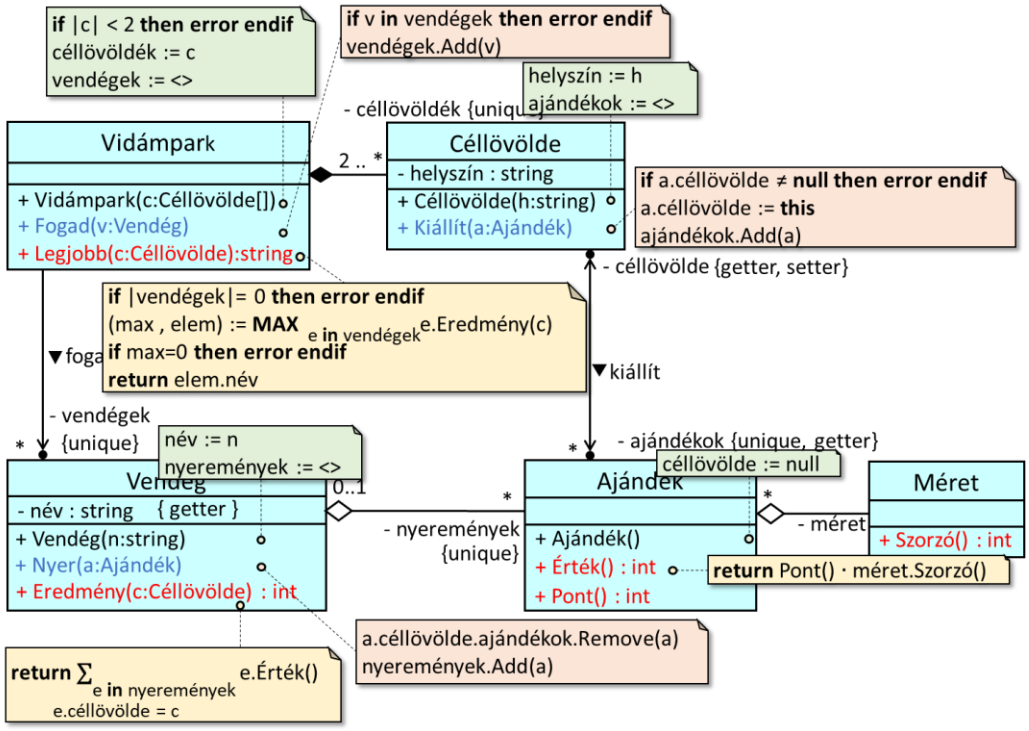
   Kezdjük most is egy használati eset diagrammal.



   A használati esetek három féle objektum felelősségi körébe sorolható: vidámpark, céllövölde, vendég. Ezeken kívül önálló objektumként jeleníthető meg az ajándék. Az objektum diagramnak érdekes része a három különböző típusú objektum kört alkotó kapcsolata.



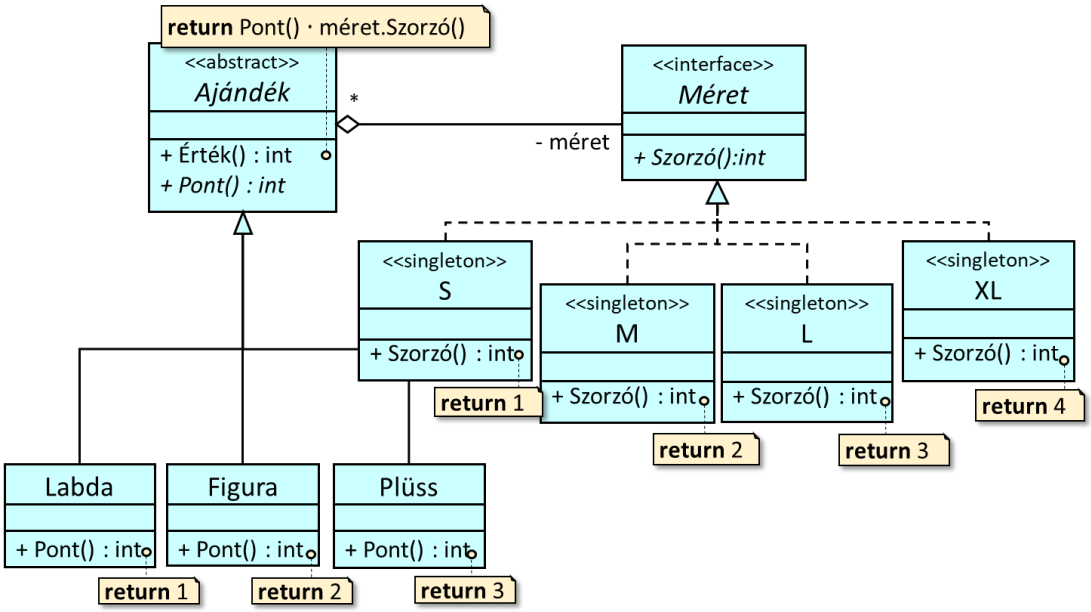
Az objektum diagramnak megfelelő osztály diagram:



Az asszociációk navigálási iránya már a megoldandó részfeladatokra vannak tekintettel. Az ajándékok gyűjtemény „unique” tulajdonságát a Kiállítja() metódus biztosítja, a nyeremények hasonló tulajdonságát a Nyer() metódus. lőtték. Egy vendég többször is ellátogathat egy céllövöldébe, ezért a vendégek gyűjtemény nem unique.

Az osztály diagramnak fontos részei a kapcsolatok létrehozásáért felelős metódusok. A Vidámpark konstruktora gondoskodik arról, hogy legalább két céllövöldéje legyen, de nem részletezzük, hogy ezeket honnan teremti elő, hiszen a feladat kérdéseinek megválaszolásához valójában nincs szükség a vidámpark objektumra. A Céllövölde Kiállít() metódusa egy új ajándékot vesz fel a céllövölde ajándékai közé (ez az ajándékok nevű gyűjtemény), amelyen feltünteti a céllövöldét (lásd céllövölde szerepnevet), hogy lehessen látni az ajándékról, hogy melyik céllövöldében lehet (vagy lehetett) elnyerni. A már elnyert ajándékról is leolvasható majd, hogy melyik céllövöldében nyerték, annak ellenére, hogy már nem lesz benne az ajándékok gyűjteményben. A Vendég Nyer() metódusa írja le azt, amikor egy céllövöldében egy ajándékot nyer a vendég. Ekkor az ajándék kikerül a céllövölde ajándékai közül, és bekerül a vendég nyereményei közé, miközben az ajándék „őrzi annak emlékét”, hogy melyik céllövöldében lőtték.

Egy céllövölde legjobb vendégét megkereső maximum kiválasztás (ez a Céllövölde osztály Legjobb() metódusa) számára fel kell sorolni a céllövöldénél regisztrált vendégeket (ugyanazt a vendéget esetleg többször is), és ki kell számolni az adott céllövöldében szerzett nyereményeiknek összértékét. Ez utóbbit a Vendég Eredmény() metódusa végezi, amely egy feltételes összegzés, ahol egy vendég adott céllövöldében nyert nyereményeit kell felsorolni, és azok érékét összeadni. Egy nyeremény (azaz ajándék) értéke (Érték()) az ajándék fajtájának pontszámától (Pont()) és a méretének szorzótényezőjétől (Szorzó()) függ.



Az Ajándék osztály Érték() metódusának kialakítását több nevezetes tervezési minta támogathatja.

Az Érték() metódus legyen egy **sablonfüggvény**, amely törzsében használt Pont() függvény absztrakt, amit az Ajándék osztályból származtatott konkrét ajándékfajták osztályai (Labda, Figura, Plüss) írnak felül.

Egy ajándék értékének kiszámolásához az ajándék méretétől függő szorzótényezőre is szükség van, amelyet **stratégia tervmintával** adunk meg. Ehhez vezettük be a Méret osztályt, amely a Szorzó() metódust specifikálja. Mivel többféle méret is van, a Méret egy tisztán absztrakt ősosztály, egy ún. interfész lesz, és ezt a konkrét méreteket leíró (S, M, L, XL) **egyke** osztályok különféle módon implementálják. Az Ajándék osztályba egy konkrét Méret típusú objektumot kell aggregálnunk (befecskendeznünk) ahhoz, hogy hozzájussunk a méretre jellemző szorzótényezőhöz.

A konkrét méret osztályok egykeként implementálják a Méret interfészt, hiszen egy méretből elég egy példány, amelyet minden olyan ajándék meghivatkozhat majd, amelynek ez a mérete.

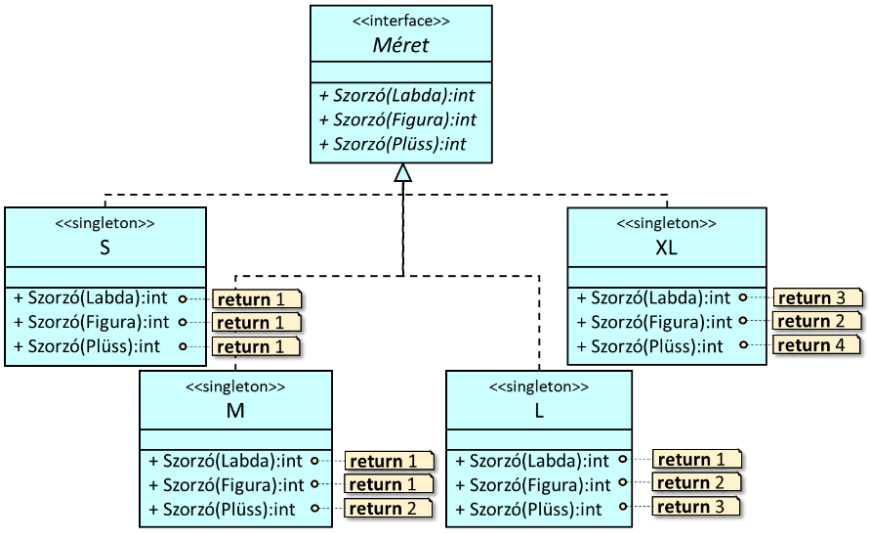
A sablonfüggvény- és a stratégia tervminta a rugalmas bővítés lehetőségét, az Open-Closed szoftvertervezési elvet támogatja, míg az egyke tervminta a memória spórolás eszköze.

* * *

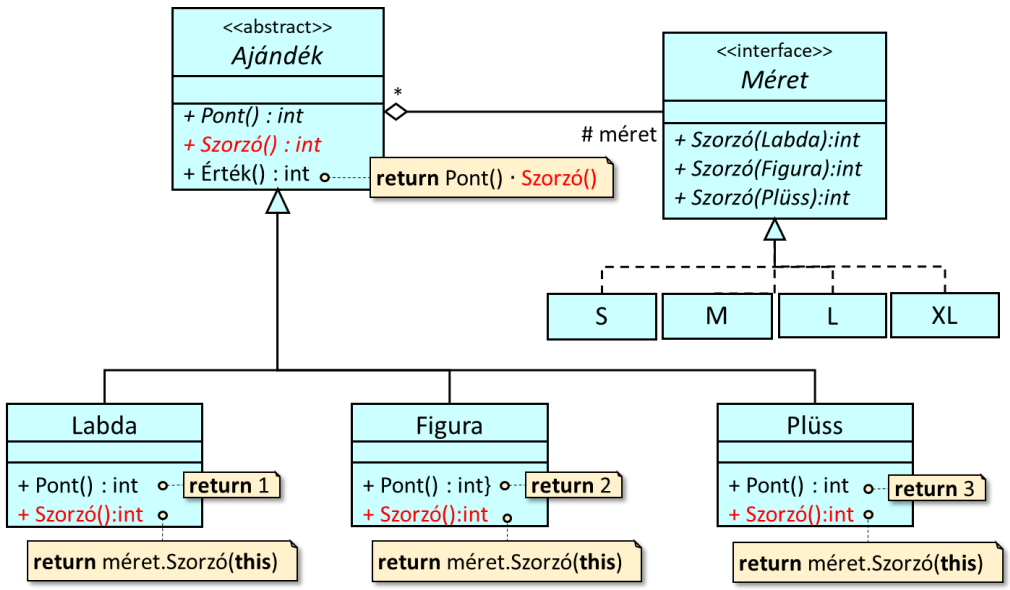
Hogyan változtassunk a modellen akkor, ha egy ajándék értékét kialakító szorzótényező nem csak az ajándék méretétől, hanem a fajtájától is függ?

	S	M	L	XL
Labda	1	1	1	3
Figura	1	1	2	2
Plüss	1	2	3	4

Ilyenkor a Méret interfészt megvalósító osztályoknak annyi Szorzó() metódusa lesz, ahányféle ajándék van (esetünkben három-három). Ezeket a metódusokat a paraméterük típusa különbözteti meg egymástól.



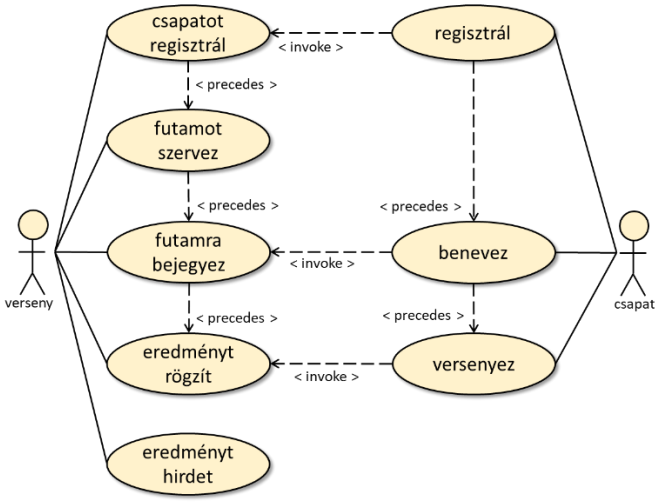
Az Ajándék osztály Érték() metódusa továbbra is Pont() · Szorzó() képlettel számolja ki egy ajándék értékét, de itt már a Szorzó() metódus is Ajándék osztálynak az absztrakt metódusa. Ezt a konkrét ajándékfajta osztályok definiálják úgy, hogy a stratégia tervmintához hasonlóan az Ajándék osztály méret szerepnevével „fecskendezik be" a mérettől és ajándékfajtától függő Szorzó() metódust, amelynek paramétere a **this** (ennek típusa adja meg a konkrét ajándék fajtáját). A méret.Szorzó(**this**) hívásakor a **this** „megy el látogatóba" a méret szerepnév konkrét osztályához, hogy a megfelelő (nemcsak a mérettől, hanem az ajándék fajtájától is függő) Szorzó() metódus eredményét kapjuk meg. Ez az úgynevezett **látogató tervezési minta**. Ez a megoldás maximálisan kielégíti a SOLID elveket.



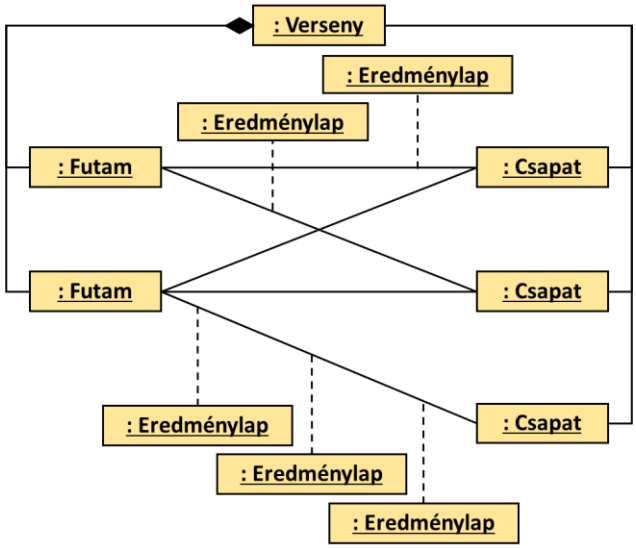
3. Egy országos rally autóversenyre történő regisztrációkor a csapatok egyedi azonosítót kapnak, amellyel az egyes futamokra benevezhetnek. Egy verseny több (legalább egy) futamból áll, egy futamon több (legalább kettő) csapat is indul legalább egy kategóriában (sportautó, teherautó, motor). A futamok megadott időben indulnak. Egy csapatnak egy futamon elért eredménye az egyes kategóriákban elért helyezéseitől függ. Ehhez mindazon kategóriában, ahol indított versenyzőt egy csapat, a *(futamon induló csapatok száma+1–helyezés)·tényező* képlet értékét kell kiszámolni, és ezen értékeket összeadni. A képletbeli tényező kategóriánként eltérő: ez a motoroknál 1, a sportautóknál 3, a teherautóknál 4.

   Melyik csapat a nyertese a teljes versenynek a futamokon elért eredményeik értékének összege alapján?

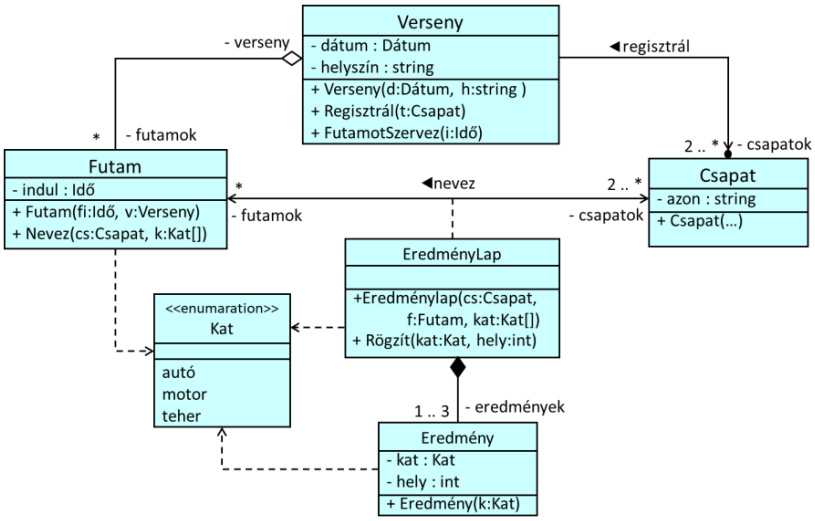
   A megoldásnak két féle aktora van: a verseny-szervező és a csapatok.



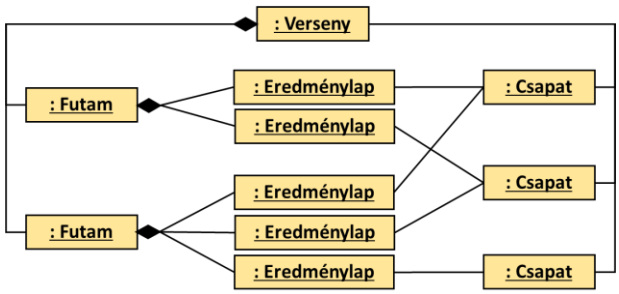
   Célszerű a feladatot verseny, csapat, és futam objektumok segítségével modellezni, ahol a futamok a verseny részei, a csapatok pedig a versenyhez, és annak néhány futamához kapcsolódnak.



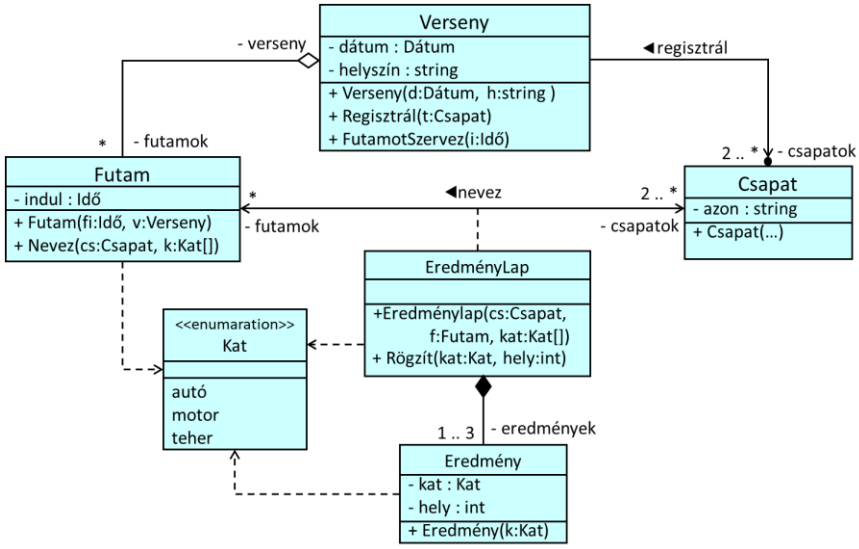
Azt is ábrázolnunk kellene valahogy, hogy egy futamra benevezett csapat mely kategóriákban indul a futamon, és abban milyen helyezéseket ért el. Ezt egy futam-csapat kapcsolathoz rendelt eredménylap objektum tartalmazhatná. Ezt osztálydiagram szinten egy EredményLap asszociációs osztállyal modellezhetjük. Minden eredménylap tartalmazza az adott csapat adott futamra benevezett kategóriáiban elért eredményeit.



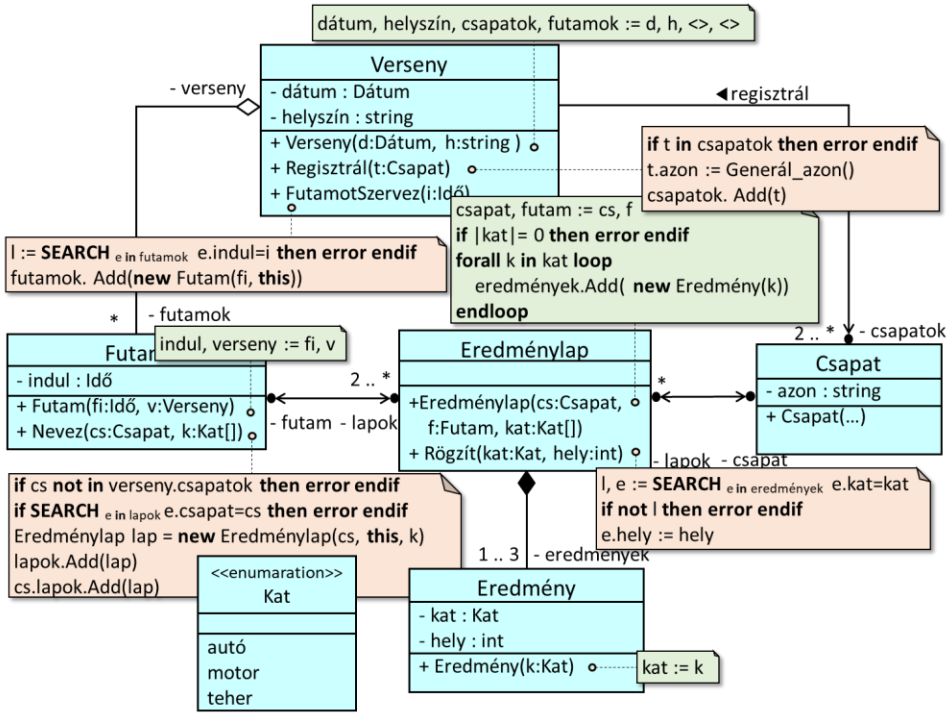
A konkrét megvalósításban az eredménylap egy olyan objektum lesz, amely összeköt egy futamot egy csapattal: azt a futamot, amelyre a csapat benevezett.



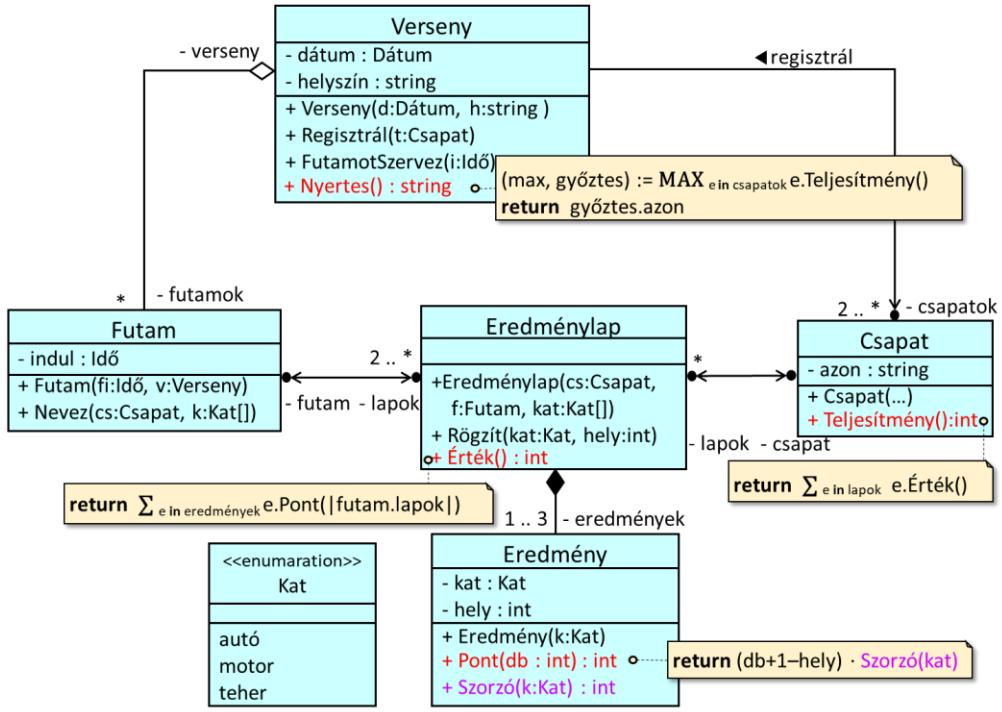
A korábban mutatott osztálydiagramnak elkészíthetjük azt a változatát, amelyben már közönséges osztályként ábrázoljuk az EredményLap osztályt.



A kompozíciós kapcsolatokat a tartalmazó osztályok konstruktorai hozzák létre, az asszociációs kapcsolatokat speciális metódusok alakítják ki.



A többi metódus a kérdésre adott választ felelős: ez a csapatok felsorolására épülő maximum kiválasztásba ágyazott dubla összegzés, ahol az összegzések felsorolják egy csapat összes futamának teljesítményét, azon belül az összes kategóriában elért eredményt.



A kategóriánkénti eredmény szorzótényezőjét származtatás segítségével specializáljuk (sablonfüggvény tervminta).