

2. Követelmény, projekt, funkcionálitás

5 – runtime_error

Konzulens:
Dobos-Kovács Mihály

Csapattagok

Mizser Ádám Zoltán	SHKGZW	mizser.adam@gmail.com
Tepliczky Olivér	WB6LC5	tepliczkyo@edu.bme.hu
Fekete Álmos Valér	KR5WPC	feketealmos0@gmail.com
Váradi Kristóf	BP17IB	kristofvaradi@edu.bme.hu
Sasvári Szabolcs Attila	TWOZG6	szabolcs.attila.sasvari@edu.bme.hu

2023.03.13.

2. Követelmény, projekt, funkcionalitás

2.1 Bevezetés

2.1.1 Cél

A dokumentum célja rögzíteni a szállítandó termék specifikációját és követelményeit, megismertetni az olvasót a projekt fejlesztése és átadása során használt (nem technikai) kulcsfogalmakkal és terminológiával, magas szintű képet szolgáltatni az elkészítendő szoftvertermék architektúrájáról és fejlesztési tervéről.

2.1.2 Szakterület

Az elkészült termék kategorikusan az Indie stratégiai videojátékok közé sorolható. A szoftver szórakoztatáipari üzleti célokban szolgál, fő célcsoportok olyan személyek és kisebb csoportok, akik érdeklődnek a stratégiai játékok és az erőforrás-gazdálkodás iránt, emellett azonban a játék oktatási célokra is felhasználható, például az erőforrás-gazdálkodás és a fenntarthatóság, vagy a játékelmélet elveinek szemléltetésére.

2.1.3 Definíciók, rövidítések

Definíció	Magyarázat
<i>Model (modell)</i>	Egy valós vagy hipotetikus világ (a „rendszer”) egy részének egyszerűsített képe, amely a rendszert helyettesíti bizonyos megfontolásokban. Egy modell elkészítésének mindenkor a kérdés megválaszolása a célja.
<i>View (nézet)</i>	A modell, és a modellhez kapcsolható folyamatok vizuális reprezentációja.
<i>Controller (kontroller)</i>	A modell elemeinek, és az ahhoz kapcsolható folyamatok logikáját és állapotát manipuláló komponens.
<i>MVC architektúra</i>	Olyan szoftverarchitektúra, amely magas absztrakciós szinten a programot a Model-View-Controller részekre osztja.
<i>Indie videojáték</i>	Független videojáték - olyan innovatív videojáték, amelyet kis szoftverfejlesztői csapat alkot meg, nagymértékű anyagi- és vállalati háttér hiányában.
<i>Funkcionális specifikáció</i>	A szoftver vagy rendszer funkcionálisának rögzítése, azaz milyen műveleteket kell elvégeznie, hogyan kell azokat végrehajtani, és milyen eredményeket kell produkálnia.

2.1.4 Hivatkozások

Megnevezés	Hivatkozás
Információk a megrendelőtől és a CTO-tól	https://www.iit.bme.hu/targyak/BMEVIIIB02
Határidők	https://www.iit.bme.hu/targyak/BMEVIIIB02%C3%BCtemterv-hat%C3%A1rid%C5%91k
A projekt GitHub repository-ja	https://github.com/RuntimeError-BME/coursework

(UML) diagram rajzoló webalkalmazás	https://app.diagrams.net/
A célcsoport kliens számítógépét szimuláló VM	https://niif.cloud.bme.hu/
Anyagok eljuttatása a megrendelőnek és CTO-nak	https://devil.iit.bme.hu:9181/hercules/start

2.1.5 Összefoglalás

A dokumentum további részének célja az, hogy átfogó képet nyújtsan a tervezett játék felépítéséről és működéséről, valamint az, hogy az olvasó megismerje projektmenedzsment és a megvalósítás tervét. A dokumentum kitér az egyes szerepek és logikai komponensek, valamint a játék folyamatának részleteire.

Az olvasó betekintést kaphat az egyes szereplők (játékosok) által végzett tevékenységek lefolyásába.

A magasszintű áttekintés és a szoftver funkcionális specifikációja mellett kitérünk a fejlesztést (és a terméket) befolyásoló körülményekre és korlátozásokra, foglalkozunk a termék funkcionalitásához, az erőforrásokhoz és az átadáshoz köthető, valamint egyéb követelményekkel is.

Megadjuk a termék főbb használati eseteinek grafikus, illetve szöveges leírását, kitérünk a megrendelők (és egyéb kiulcső személők) számára lényeges definíciókra és kulcskifejezésekre.

Részletes képet adunk a projekt realizációjának lépéseiiről, a tervezett határidőkről, valamint a megvalósításhoz használt eszközökről, erőforrásokról, technológiákról és módszerekről, és magáról a folyamatszervezésről is.

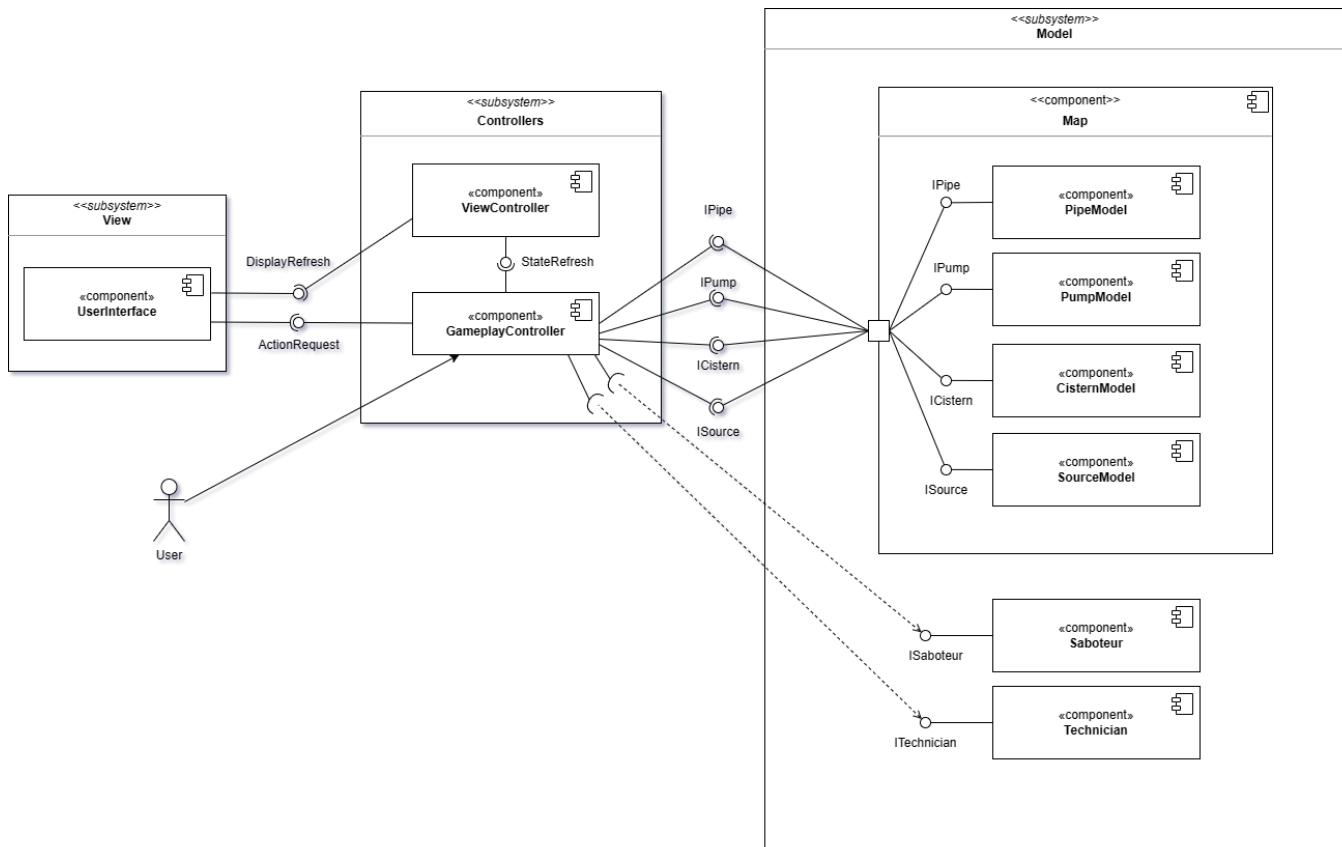
2.2 Áttekintés

2.2.1 Általános áttekintés

2.2.1.1 Komponensek áttekintése

A szoftvertermék – a legfelsőbb szinten – az MVC architektúra szerint három alrendszerre osztható, ezek a *Model*, *View* és a *Controller*(-s) alrendszerök. A *View* alrendszer felelőssége a felhasználók felé vizuálisan közölni a játék állapotát, valamint kommunikálni a játékok vezérlő komponensekkel (a kontrollerekkel).

A *Controllers* alrendszer a legfelsőbb absztraktiós szinten két részre oszthatjuk. A *ViewController* utasításokat ad a *View*-nak a vizuális megjelenítés frissítésére, és kommunikál a *GameplayController*-rel. A *GameplayController* fő felelőssége a rendszer állapotának változtatása és karbantartása, valamint a felhasználókkal való kommunikáció megvalósítása. A *GameplayController* kapcsolódik a *Model* alrendszerhez; a *Map* összetett komponensből, és a *Saboteur*, valamint *Technician* komponensekből áll. A *Map* a pálya (értsd: Drukmáori Sivatag) modellje, részei pedig a sivatagban megtalálható – modellezési szempontból lényeges – objektumok. A *Model* alrendszerben megtalálható a *Saboteur* és *Technician* komponens is, amelyek a sivatagban tevékenykedő „személyek” modellbeli reprezentációi.



2.2.1.2 Interfészek felelősségei

View: A *UserInterface* komponens a *DisplayRefresh* interfészt szolgáltatja a *Controllers* alrendszer *ViewController* komponense felé, amely lehetőséget ad a játék állapotának frissítésére (módosítására).

ViewController: A *ViewController* a *StateRefresh* interfészt szolgáltatja a *GameplayController* komponensnek, a két kontroller ennek segítségével kommunikál, és a *GameplayController* így közli a játékmenet és a pálya változásait a *ViewController*-rel.

GameplayController: a *GameplayController* az *ActionRequest* interfészt szolgáltatja a *UserInterface* komponens felé, amellyel az értesíteni tudja a grafikus program kéréseiről (pl.: állapotfrissítés kérése, cselekvés kérése a grafikus felületen történt változások miatt).

PipeModel, PumpModel, CisternModel, SourceModel: a pályamodell komponensei interfészeket szolgáltatnak a *GameplayController* felé, amelyekkel a játékmenet során az változtatni tudja állapotukat, és lekérdezheti állapotukat.

Saboteur, Technician: a szabotőr, ill. szerelő modellje interfészt szolgáltat a *GameplayController* felé, mellyel az a játékmenet során „cselekvésre” tudja ösztökelníi őket, leolvashatja állapotukat, és kommunikálhat velük.

2.2.2 Funkciók

Megjegyzés: a fejlesztőcsapatunk által hozzáírt funkciókat és pontosításokat aláhúzással jelöljük

A drukmákorú sivatagon át bonyolult csőrendszer szállítja a vizet a hegyi forrásokból a sivatagon túl elterülő városok ciszternáiba. A csőrendszer egyszerű, elágazás nélküli csövekből és a csövekhez csatlakozó aktív elemekből (forrás, ciszterna, napelemmel működő vízátemelő pumpa stb.) áll. Egy pumpa több (de a pumpára jellemző véges számú) csövet is összeköthet, és minden pumpán külön-külön állítható, hogy éppen melyik belekötött csőből melyik másik csőbe pumpáljon, azonban egyszerre csak egy bemenete és egy kimenete lehet. A többi rákötött cső eközben el van zárva. A

pumpák véletlen időközönként el tudnak romlani, ilyenkor megszűnik az adott pumpánál a vízáramlás. A pumpák mindegyike rendelkezik egy víztartályval, amit a víz átemelése közben használ átmeneti tárolóként. A pumpa csak akkor tud vizet pumpálni egy csőbe, ha a cső szabad kapacitása ezt lehetővé teszi.

A csőhálózat bővíthető, változtatható. A csövek kellően rugalmasak ahhoz, hogy az egyik végüköt lecsatlakoztatva egy másik aktív elemhez vagy csőhöz elvihetők és ott felcsatlakoztathatók legyenek. A ciszternáknál folyamatosan készülnek az új csövek, amelyek egyik vége a ciszternához kapcsolódik, a másik azonban szabad. A szabad végű csövekből a csőbe betáplált víz a homokba folyik.

A játék felülnézetes. A fix méretű pálya négyzetrácsos, és négyzet alakú elemekből áll. A sivatagban van egy meglévő csőhálózat a játék kezdetén, ami minden kezdéskor ugyanaz.

A városban élő munkások, akik a csöveget gyártják, nem a legokosabbak. Az elkészült csöveget a ciszternájuktól közyetlenül északra helyezik el. Ha ez nem lehetséges (mert van ott már cső), akkor nyugatra próbálják, ha ez sem, akkor keletre, végül pedig délről próbálkoznak. Amikor nincs szabad hely egyik irányba sem a ciszternáktól, nem gyártanak új csövet.

A csőhálózatot a szerelők tartják karban. Ők javítják meg az elromlott pumpákat, ők állítják át a pumpákat, hogy minden lehetséges legtöbb víz tudjon áthaladni a hálózaton, és ha egy cső kilyukad, az ő dolguk a cső megfoltolása is. A szerelők úgy tudnak elemeket megjavítani, ha ott tartózkodnak rajtuk. A kilyukadt csövekből a víz kifolyik, a csövek végén lévő pumpához már nem jut belőle. A szerelők dolga a ciszternáknál lévő szabad csövekkel a hálózat kapacitásának növelése. A szerelők képesek a mellettük lévő csöveket magukhoz venni, ha nem tartózkodik rajtuk senki, de egyszerre csak egy lehet náluk. Ezt elhelyezhetik úgy, hogy egy csőnek legfeljebb 2 szomszédja lehet (nem lehetnek párhuzamos csövek), az aktív elemeknek pedig 4 (de ez csak cső szomszédokkal megoldható). A szerelők a ciszternáknál magukhoz tudnak venni új pumpát is, amit egy cső közepén tudnak elhelyezni, de be kell tartaniuk azt a szabályt, hogy két aktív elem sosem lehet szomszédos. A csövet ehhez ketté kell vágni, és a két végét a pumpához kell csatlakoztatni. Ehhez el kell vinniük magukkal a pumpát a megfelelő helyre. Ilyenkor az eredeti cső elvész, és helyére kerül a pumpa. Egyszerre csak egy elem lehet egy szerelőnél: vagy cső, vagy egy pumpa, és csak maguk melletti helyekre képesek letenni őket.

A hálózaton élnek a nomád szabotörök is, akik a pumpákat tudják átállítani és a csöveket szokták kilyukasztani, amennyiben rajtuk állnak.

Mivel a sivatag veszélyes hely, a szerelők és a szabotörök csak a csőhálózaton haladhatnak. A pumpáknál kikerülhetik egymást, de a csöveken már nem tudnak elmenni egymás mellett, egy csövön egyszerre csak egy ember állhat. Forrásokon és ciszternákon is állhatnak többen.

A játékot a két csapat legalább 2-2 játékossal játszsa, és legfeljebb 3-3. A szabotörök dolga, hogy minél több víz folyjon el a lyukakon, a szerelők pedig azon dolgoznak, hogy minél több víz jusson a ciszternákba. Az a csapat nyer, amelyik a játék végére több vizet szerez. A játéknak akkor van vége, amikor az egyik csapat megszerez egy bizonyos víz mennyiséget. A forrásokból mind a 4 irányból jön a víz, a ciszternákba pedig minden a 4 irányba érkezhet. Ha több irányból történnék ezek, akkor az több pontnak számít a megfelelő csapatnak.

A játék körökből (round) áll. minden kör elején minden ciszternához terem egy új cső, a fent leírtak szerint, ha ez lehetséges. A szerelők és a szabotörök felváltva kerülnek sorra, előre meghatározott, állandó sorrendben. Egy játékos tetszőleg lépést megtehet (akkár helyben is maradhat), de csak egy interakciót hajthat végre, amivel véget is ér a köre (turn), de interakció nélkül is lezárhatja a körért. Amikor egy játékosnak megkezdődik a köre, akkor folyik víz, és akkor kapnak pontokat a csapatok, illetve a pumpák is ilyenkor romolhatnak el. A ciszternákat biztonságos utak kötik össze, ezért átjárhatók úgy is, hogy nincsenek csővel összekötve. Amíg egy játékos köre tart, addig a többi játékos nem tud interaktálni a játékkal.

2.2.3 Felhasználók

A program felhasználónak nem kell magas szintű informatikai háttér tudással rendelkezni, illetve a program használatához, elindításához, elegendő az alap számítógépes ismeret. A programot egyszerre több felhasználó is használhatja, viszont egy időben egyszerre csak egy tudja kezelni.

2.2.4 Korlátozások

Elvárás, hogy a leírt, előre meghozott szabályok szerint lehessen a játékot játszani, illetve, hogy a program betartassa ezeket a szabályokat a felhasználókkal/játékosokkal. Továbbá elvárás, hogy a program ne fagyjon le, crasheljen ki, dobjon ki hibakódokat játék közben, illetve ne hajson végre olyan műveleteket, amik a megrendelő által támasztott követelményektől merőben eltérnének.

A játék játszásához minimum 2-2, azaz 4 játékos szükséges.

2.2.5 Feltételezések, kapcsolatok

A dokumentum „2.1.4 Hivatkozások” szekciójában, a hivatkozások táblázat megnevezések oszlopá szolgál a web-oldalak és a feladat kapcsolatának meghatározására.

2.3 Követelmények

2.3.1 Funkcionális követelmények

Rövidítések:

Ellenőrzés: S, P, G, B: skeleton, prototípus, grafikus, beadás

Prioritások: A, F, O: alapvető, fontos, opcionális

Forrás: M, F: megrendelő, fejlesztőcsapat

Azonosító	Leírás	Ellenorzés	Prioritás	Forrás	Use-case
D01	A csőrendszer egyszerű, elágazás nélküli csövekből és a csövekhez csatlakozó aktív elemekből (forrás, ciszterna, napelemmel működő vízátemelő pumpa stb.) áll.	P G B	A	M	View Pipenetwork
D02	Egy pumpa több (de a pumpára jellemző véges számú) csövet is összeköthet. A pumpák mindegyike rendelkezik egy víztartályval, amit a víz átemelése közben használ átmeneti tárolóként. A pumpa csak akkor tud vizet pumpálni egy csőbe, ha a cső szabad kapacitása ezt lehetővé teszi.	P G B	A	M	Control Waterflow View Pipenetwork
D03	A pumpák véletlen időközönként el tudnak romlani, ilyenkor megszűnik az adott pumpánál a vízáramlás.	G B	F	M	Impair Pump Control Waterflow View Pipenetwork
D04	Minden pumpán külön-külön állítható, hogy éppen melyik belekötött csőből melyik másik csőbe pumpáljon, azonban egyszerre csak egy bemenete és egy kimenete lehet. A többi rákötött cső eközben el van zárva.	P G B	A	M	Change Pump direction Control Waterflow View Pipenetwork
D05	A csövek kellően rugalmasak ahhoz, hogy az egyik végüköt lecsatlakoztatva egy másik aktív elemhez elvihetők és ott felcsatlakoztathatók.	G B	F	M	Relocate Pipe Store Part View Pipenetwork

D06	A ciszternáknál folyamatosan készülnek az új csövek, amelyek egyik vége a ciszternához kapcsolódik, a másik azonban szabad.	G B	F	M	Produce Part View Pipenetwork
D07	A szabad végű csövekből a csőbe betáplált víz a homokba folyik.	P G B	A	M	Control Waterflow View Pipenetwork
D08	A csőhálózatot a szerelők tartják karban. Ők javítják meg az elromlott pumpákat, ők állítják át a pumpákat, hogy minden lehetséges legtöbb víz tudjon áthaladni a hálózaton, és ha egy cső kilyukad, az ő dolguk a cső megfeszítése is.	P G B	A	M	Repair Part Change Pump direction View Pipenetwork
D09	A kilyukadt csövekből a víz kifolyik, a csövek végén lévő pumpához már nem jut belőle.	G B	A	M	Control Waterflow View Pipenetwork
D10	A szerelők a ciszternáknál magukhoz tudnak venni új pumpát is.	G B	F	M	Store Part View Pipenetwork
D11	A pumpákat a cső közepén tudják a szerelők elhelyezni. A csövet ehhez ketté kell vágni, és a két végét a pumpához kell csatlakoztatni.	G B	F	M	Place Pump View Pipenetwork
D12	A hálózaton élnek a nomád szabotörök is, akik a pumpákat tudják átállítani, és a csöveket szokták kilyukasztani.	P G B	A	M	Hole Pipe Change Pump direction View Pipenetwork
D13	Mivel a sivatag veszélyes hely, a szerelők és a szabotörök csak a csőhálózaton haladhatnak. A pumpáknál kikerülhetik egymást, de a csöveken már nem tudnak elmenni egymás mellett, egy csövön egyszerre csak egy ember állhat.	S P G B	A	M	Move Player View Pipenetwork
D14	A játékot a két csapat legalább 2-2 játékossal játsza.	G B	A	M	View Pipenetwork
D15	A szabotörök dolga, hogy minél több víz folyjon el a lyukakon, a szerelők pedig azon dolgoznak, hogy minél több víz jusson a ciszternákba. Az a csapat nyer, amelyik a játék végére több vizet szerez.	P G B	A	M	View Pipenetwork
D16	A játék felülnézetes. A fix méretű pálya négyzetrácsos, és négyzet alakú elemekből áll.	P G B	A	F	View Pipenetwork
D17	A sivatagban van egy meglévő csőhálózat a játék kezdetén, ami minden kezdéskor ugyanaz.	G B	A	F	View Pipenetwork
D18	Az elkészült csövek a ciszternájuktól közvetlenül északra lesznek elhelyezve. Ha ez nem lehetséges (mert van ott már cső), akkor nyugatra, ha ez sem, akkor keletre, végül pedig délre. Amikor nincs szabad hely egyik irányba sem a ciszternáktól, nem terem új cső körülötte.	G B	F	F	Produce Part View Pipenetwork
D19	A szerelők és a szabotörök akkor tudnak interaktálni a part-akkal, ha rajtuk állnak.	P G B	A	F	Repair Part Change Pump direction Hole Pipe View Pipenetwork
D20	A szerelők képesek a mellettük lévő csöveket magukhoz venni, ha nem áll rajtuk senki.	G B	F	F	Store Part View Pipenetwork

D21	A szerelők a náluk lévő csövet elhelyezhetik úgy, hogy egy csőnek legfeljebb 2 szomszédja lehet (nem lehetnek párhuzamos csövek). Az aktív elemeknek 4 szomszédjuk lehet max.	G B	F	F	Relocate Pipe Store Part View Pipenetwork
D22	A szerelőknek el kell vinniük magukkal a pumpát a megfelelő helyre, hogy le tudják rakni. Ilyenkor az eredeti cső elvész, és helyére kerül a pumpa. Két aktív elem sosem lehet szomszédos.	G B	F	F	Place Pump Store Part View Pipenetwork
D23	A szerelők csak maguk melletti helyekre képesek part-okat letenni.	G B	F	F	Relocate Pipe Place Pump View Pipenetwork
D24	A szerelőknél egyszerre csak egy part lehet.	G B	O	F	Store Part View Pipenetwork
D25	Forrásokon és cisternákon is állhatnak többen.	S P G B	A	F	Move Player View Pipenetwork
D26	A játékot a két csapat legfeljebb 3-3 játékkossal játsza.	G B	O	F	View Pipenetwork
D27	A játéknak akkor van vége, amikor az egyik csapat megszerez egy bizonyos víz mennyiséget.	P G B	A	F	Control Waterflow
D28	A forrásokból mind a 4 irányból jön a víz, a cisternákba pedig mind a 4 irányba érkezhet. Ha több irányból történnek ezek, akkor az több pontnak számít a megfelelő csapatnak.	G B	F	F	Control Waterflow View Pipenetwork
D29	A játék körökből (round) áll. minden kör elején minden cisternához terem egy új cső. A szerelők és a szabotörök felváltva kerülnek sorra, előre meghatározott, állandó sorrendben.	G B	A	F	Produce Part View Pipenetwork
D30	Egy játékos a körében (turn) tetszőleg lépést megtehet (akár helyben is maradhat), de csak egy interakciót hajthat végre, amiivel véget is ér a köre, de interakció nélkül is lezárhatja azt. Amíg egy játékos köre tart, addig a többi játékos nem tud interaktálni a játékkal.	P G B	A	F	Move Player Cross Cisterns Store Part Repair Part Change Pump direction Relocate Pipe Place Pump Hole Pipe View Pipenetwork
D31	Amikor egy játékosnak megkezdődik a köre (turn), akkor folyik víz, és akkor kapnak pontokat a csapatok, illetve a pumpák is ilyenkor romolhatnak el.	P G B	A	F	Control Waterflow Impair Pump View Pipenetwork
D32	A cisternákat biztonságos utak kötik össze, ezért átjárhatók úgy is, hogy nincsenek csővel összekötve.	G B	O	F	Cross Cisterns View Pipenetwork

2.3.2 Erőforrásokkal kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
E01	<i>A forrásprogramnak a kari felhőben (https://niif.cloud.bme.hu/) biztosított környezetben, az ott megtalálható JDK alatt (parancssorból) lefordíthatónak és futtathatónak kell lennie.</i>	B	A	M	

2.3.3 Átadással kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
A01	<i>A program forráskódját a termék célközönsége számára a http://www.iit.bme.hu/hercules oldalon keresztül elérhetővé, azt onnan letölthetővé kell tenni.</i>	S P G B	A	M	
A02	<i>A szoftver dokumentációját és egyéb segédanyagait nyomtatott anyagként, valamint az A01 követelményben említett oldalon keresztül elérhetővé kell tenni.</i>	S P G B	A	M	
A03	<i>A szoftver átadásakor jelen lévő fejlesztők felkészülten érkezzenek. A tesztelő felhasználó kérdéseire hétköznapi nyelven tudjanak válaszolni.</i>	B	A	M	
A04	<i>A technikailag nem képzett felhasználó a lefordított tárgyprogram birtokában a segédletek segítségével képes önmagától üzembe helyezni a programot.</i>	B	A	M	

2.3.4 Egyéb nem funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
F01	<i>A program manuális tesztelés esetén a dokumentációban leírt és a követelmények által elvárt eredményeket produkálja.</i>	B	A	M	

2.4 Lényeges use-case-ek

2.4.1 Use-case leírások

Use-case neve	Move Player
Rövid leírás	<i>A játékosok a szabotőröket és szerelőket mozgatják a csőhálózaton.</i>
Aktorok	Player
Főforgatókönyv	<ol style="list-style-type: none"> 1. A Játékos saját figuráját lépteti egy csőről egy másik szomszédos csőre. 2. A Játékos saját figuráját lépteti egy csőről egy szomszédos aktív elemre. 3. A Játékos saját figuráját lépteti egy aktív elemről egy választott szomszédos csőre.
Alternatív forgatókönyv	1.A.1. / 3.A.1. A játékos nem tud csőre lépni, mert már foglalt.

Use-case neve	Change Pump direction
Rövid leírás	<i>A játékos változtatja, hogy a pumpa melyik csőből melyikbe engedje a vizet.</i>
Aktorok	Player
Főforgatókönyv	<ol style="list-style-type: none"> 1. A játékos elforgatja azt a pumpát, amin áll, hogy az egyik csőből egy másikba folyjon a víz.
Alternatív forgatókönyv	<p>1.A.1. Ha eddig folyt át rajta víz, akkor jelzi a forgatás előtti cél csőnek, hogy már nem kap vizet, ami át adja a következő csőnek/pumpának, hogy már nincs víz, és így tovább, amíg van következő cső, ami nem szabad végű / lyukas, vagy van következő pumpa, ami nem elromlott / zárt bemenetű az adott irányból.</p> <p>1.A.2. Ha a forgatás utáni forráscsőből folyik víz, akkor az addig folyik a következő csövekbe (akár pumpákon át), amíg nem ér szabad végű / lyukas csőhöz vagy elromlott / zárt bemenetű pumpához vagy ciszternához.</p>

Use-case neve	View Pipenetwork
Rövid leírás	<i>A játékos megtekinti a csőhálózatot.</i>
Aktorok	Player
Főforgatókönyv	<ol style="list-style-type: none"> 1. A rendszer kirajzolja a csőhálózat aktuális állapotát 2. A csapatok megtekintik a csőhálózat aktuális állapotát
Alternatív forgatókönyv	-

Use-case neve	Cross Cisterns
Rövid leírás	<i>A játékos egy ciszternából egy másikba megy.</i>
Aktorok	Player
Főforgatókönyv	<ol style="list-style-type: none"> 1. Ha a játékos egy ciszternában tartózkodik, akkor közvetlenül átutazhat egy másik ciszternába.
Alternatív forgatókönyv	-

Use-case neve	Hole Pipe
Rövid leírás	<i>A szabotőr kilyukasztja azt a csövet, amin áll.</i>
Aktorok	Saboteur
Főforgatókönyv	1. A szabotőr kilyukasztja azt a csövet, amin áll.
Alternatív forgatókönyv	<p>1.A.1. Ha folyt víz a csőbe, akkor az a sivatagba folyik a lyukasztás után.</p> <p>1.A.1.A.1. Ha eddig egy másik part-ba folyt belőle a víz, akkor lyukasztás után abban már nem lehet víz, és ha abból is tovább folyt egy másikba, akkor már abban se lehet, és így tovább, amíg van következő part, amiben víz folyt.</p>

Use-case neve	Store Part
Rövid leírás	<i>A szerelő elrak a tárolójába egy pumpát vagy egy csövet.</i>
Aktorok	Technician
Főforgatókönyv	<p>1. A szerelő megkísérel eltenni a tárolójába egy szomszédos csövet.</p> <p>2. A szerelő egy ciszternára lép, hogy magához vegyen egy pumpát.</p>
Alternatív forgatókönyv	<p>1.A.1. / 2.A.1. A szerelő tárolója üres (nincs nála olyan part, amit nem tett még le).</p> <p>1.A.1.A.1. A cső, amit magához próbál venni, nem lyukas.</p> <p>1.A.1.A.1.A.1. Nem áll játékos a csövön, amit magához próbál venni.</p> <p>1.A.1.A.1.A.1.A.1. / 2.A.1.A.1. A szerelő tárolójába kerül a part.</p>

Use-case neve	Relocate Pipe
Rövid leírás	<i>A szerelő lerakja az tárolójából a csövet egy szabad helyre.</i>
Aktorok	Technician
Főforgatókönyv	1. A szerelő megkíséri lerakni a tárolójából a csövet egy szomszédos, szabad helyre.
Alternatív forgatókönyv	<p>1.A.1. A szerelő nem olyan cső mellé próbálja letenni a csövet, aminek már van 2 szomszédja.</p> <p>1.A.1.A.1. A cső lerakásra kerül a kiválasztott helyre, és a szerelő tárolója újra üres lesz.</p>

Use-case neve	Place Pump
Rövid leírás	<i>A szerelő lerakja az tárolójába a pumpát egy szabad helyre.</i>
Aktorok	Technician
Főforgatókönyv	1. A szerelő megkíséri lerakni a tárolójából a pumpát egy szomszédos cső helyére.
Alternatív forgatókönyv	<p>1.A.1. A kiválasztott csőnek 2 cső szomszédja van.</p> <p>1.A.1.A.1. A pumpa kicsérélődik a kiválasztott csőre, és a szerelő tárolója újra üres lesz.</p>

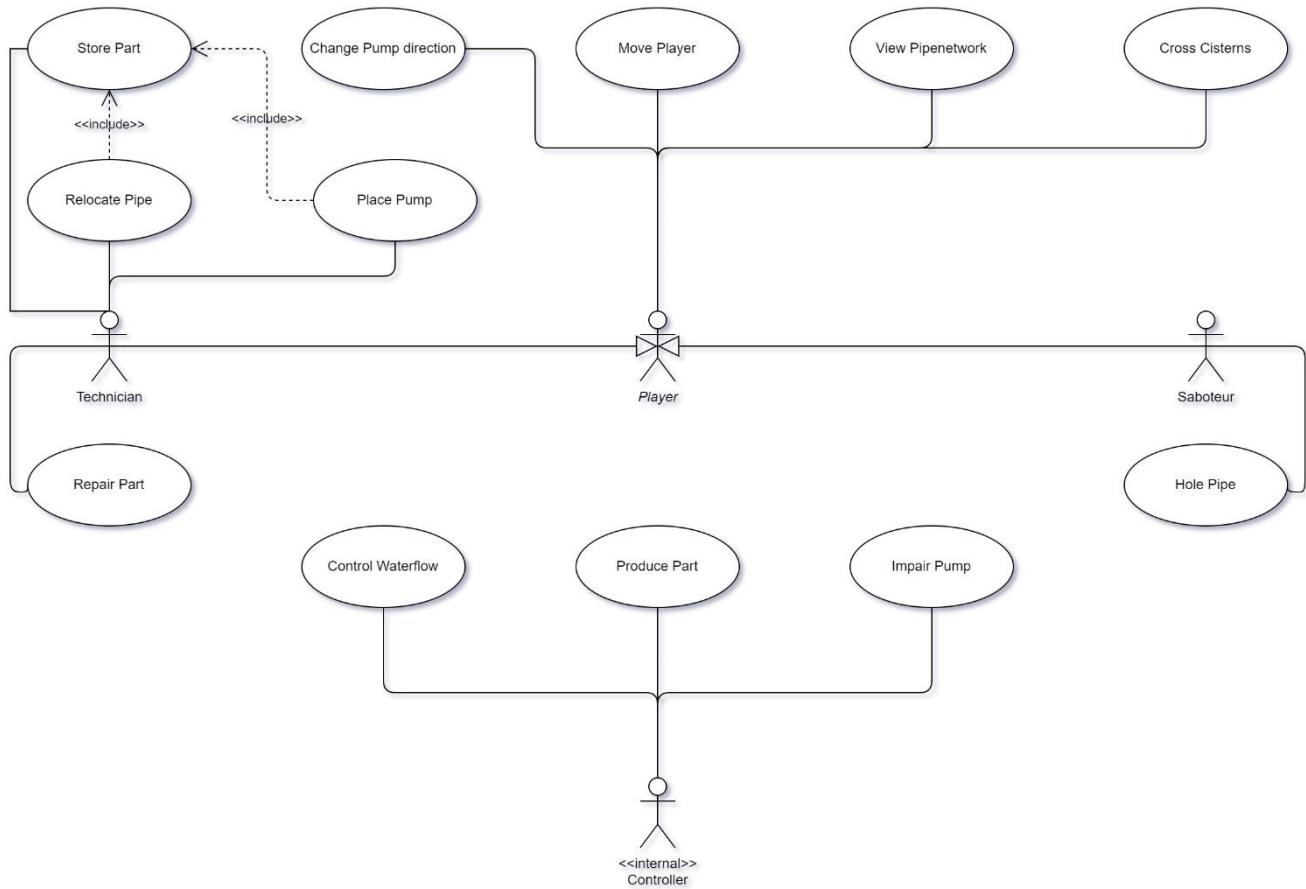
Use-case neve	Repair Part
Rövid leírás	<i>A szerelő megjavítja a pálya azon elemét, amin áll.</i>
Aktorok	Technician
Főforgatókönyv	<p>1. A szerelő egy csövet javít meg (csövön áll)</p> <p>2. A szerelő egy pumpát javít meg (pumpán áll)</p>
Alternatív forgatókönyv	<p>1.A.1. Ha a megjavított csónék van olyan szomszédja, amiből víz érkezik belé, akkor addig folyik tovább, amíg van következő cső, ami nem lyukas / szabad végű, vagy pumpa, ami nem elromlott / zárt bemenetű.</p> <p>2.A.1. Ha a megjavított pumpa forráscsövében víz van, akkor addig folyik tovább, amíg van következő cső, ami nem lyukas / szabad végű, vagy pumpa, ami nem elromlott / zárt bemenetű.</p>

Use-case neve	Control Waterflow
Rövid leírás	<i>A víz folyásában idéz elő változásokat.</i>
Aktorok	Controller
Főforgatókönyv	<p>1. Egy játékos turn-je elején a víz elfolyik addig, ameddig folyhat. (ennek részleteiért lásd a köv. use-case-eket: Change Pump direction, Hole Pipe, Relocate Pipe, Place Pump, Repair Part)</p>
Alternatív forgatókönyv	<p>1.A.1. Víz ér a sivatagba.</p> <p>1.A.1.A.1. A szabotörök csapata annyi pontot szerez, ahány helyen víz ér a sivatagba (szabad végű csöveknél, lyukas csöveknél vagy közvetlenül a források üres szomszédjainak helyein).</p> <p>1.B.1. Víz ér ciszternába.</p> <p>1.B.1.A.1. A szerelők csapata annyi pontot szerez, ahány csőből víz érkezik a ciszternákba.</p> <p>1.C.1. Ha az egyik csapat eléri a győzelemhez szükséges pontok számát, akkor véget ér a játék, és ők nyertek.</p>

Use-case neve	Produce Part
Rövid leírás	<i>Minden körben (round) minden ciszternánál terem egy új cső, ha van üres szomszédja, illetve a rálépő szerelő kaphat pumpát.</i>
Aktorok	Controller
Főforgatókönyv	<p>1. Megkezdődik egy round.</p> <p>2. Az üres tárolójú, ciszternán álló szerelők pumpát kapnak.</p>
Alternatív forgatókönyv	<p>1.A.1. A ciszternának van üres szomszédja.</p> <p>1.A.1.A.1. Pontosan egy cső terem tőle a következő irányok egyikébe: északra, nyugatra, keletre, délre. Ez prioritási sorrend is, az első irányba fog teremni, amerre üres szomszédja van.</p>

Use-case neve	Impair Pump
Rövid leírás	<i>Véletlenszerűen tönkre mennek a pumpák a csőhálózatban.</i>
Aktorok	Controller
Főforgatókönyv	<p>1. Egy játékos turn-jének az elején minden pumpán végig megy, és kis valószínűsséggel elrontja őket.</p>
Alternatív forgatókönyv	<p>1.A.1. Ha egy pumpa elromlik, nem tud víz átfolyni rajta. Az átmeneti tárolójában lévő víz megmarad, de a kimeneti csövébe már nem folyik víz (és az azt követő szomszéd part-okban sem).</p>

2.4.2 Use-case diagram



2.5 Szótár

aktív elemek: minden olyan elem, ami nem cső (pumpa, forrás vagy ciszterna). Ezeken egyszerre több játékos is tartózkodhat. A csövekkel ellentétben 2 helyett 4 szomszédjuk lehet.

ciszterna: olyan aktív elem, amelybe, ha bármikor víz jut el, akkor a szerelők pontot szereznek. minden round elején terem egy cső szomszédja tőle északra, ha még nincs neki. Ha van neki, akkor nyugatra, ha ott is van neki, akkor keletre, végső esetben pedig délre. Ha minden irányba van szomszédja, akkor nem terem új cső szomszédja. A rajta álló szerelők kapnak egy pumpát, amennyiben nincsen náluk part.

csapat: egy célért küzdő játékosok, akik közösen tudnak nyerni vagy veszíteni. 2 van belőlük: a szerelők és a szabotőrök.

cső: olyan elem, amiben víz képes folyni egy másik elembe vagy szabad vége esetén a sivatagba

cső megfoltolása: cső megjavítása

csőrendszer, csőhálózat: a játéktéren lévő csövek és pumpák összessége

elem: négyzet alakú, a négyzetrácsos játéktéren elhelyezkedő entitás, amin a játékosok állhatnak és mozoghatnak

elem elvívése/cipelése: egy szerelő mozgása (lépéseinak sorozata), amikor van nála egy part

elem megjavítása: az interakció, amely során egy elem visszakerül az alap helyzetébe. Csövek esetén azt jelenti, amikor újra képes lesz víz szállítására a sivatagba folyatás helyett. Pumpák esetén pedig azt, amikor újra képes lesz az egyik bemenetén érkező vizet kibocsátani egy kimenetén.

elem szomszédja: az adott elemtől közvetlenül egy négyzetráccsal északra, nyugatra, keletre vagy délről lévő másik elem a játéktéren

elromlott pumpa: olyan pumpa, amely nem tud vizet átereszteni magán, amíg egy szerelő meg nem javítja

forrás: olyan aktív elem, amely vizet bocsát ki minden a 4 irányba a szomszéd elemeinek vagy a sivatagba

interakció: egy játékosok által kezdeményezett tevékenység, ami hatással van a csőhálózatra, és az adott játékos körének (turn) végét eredményezi. A lehetséges interakciók: cső kilyukasztása, elem megjavítása, pumpa átállítása és part elhelyezése (cső esetén egy üres szomszédos négyzetrácsra, pumpa esetén a cső helyére, amin a szerelő éppen áll)

interaktál: elvégez egy interakciót

lyukas/kilyukadt cső: olyan cső, amelyből a befolyó víz nem a potenciálisan másik végén lévő csőbe, hanem a sivatagba folyik

lépés: a folyamat, amely során egy játékos átkerül egy elemről egy szomszédos másik elemre

magához vesz: az a folyamat, amely során egy szerelőhöz kerül egy part, ami nem lesz megjelenítve a játéktéren, amíg le nem teszi valahová bizonyos feltételek teljesülése esetén. Egyszerre csak egy dolgot tudnak magukhoz venni / maguknál tartani.

munkások: a játékban ténylegesen nem megjelenő fogalom, ami egyedül a part-ok ciszternáknál termését hivatott magyarázni

(nomád) szabotőr: olyan játékos, akinek célja, hogy minél több víz folyjon a sivatagba. Képes a pumpák átállítására, és a csövek kilyukasztására.

part: elem, amiből a szerelők képesek pontosan egyet maguknál tartani (cső vagy pumpa)

pont: csapatonként gyülemlő, a győzelemhez közelebb vivő mérhető érték. A szabotőrök csapata a víz sivatagba folyásával, a szerelők csapata pedig a ciszternákba folyásával szerzi.

pumpa: olyan aktív elem, amely a legfeljebb 4 szomszédos csöve közül juttat vizet pontosan az egyikból pontosan egy másikba. Az előbbi cső a bemenete az utóbbi pedig a kimenete. Véletlen időközönként elromlik, és ilyenkor nem tud víz átáramlani rajta.

pumpa átállítása: az interakció, amikor egy játékos egy pumpán állva megváltoztatja annak bemenetét és kimenetét.

round: a játéknak egy olyan időegysége, ami minden játékosnak egy turn-jét magába foglalja

sivatag: a négyzetráccos játéktér minden négyzetrácsa, amin nincsen elem. A lyukas csövek esetén is használt fogalom, hogy a víz a "sivatagba folyik". Ekkor a lyukas csőbe érkező víz a sivatagba kerül, nem halad tovább a csőrendszerben. Ha bármikor víz jut a sivatagba, akkor a szabotőrök pontot szereznek.

szabad kapacitású cső: olyan cső, amiben nem áramlik víz

szerelő: olyan játékos, akinek célja, hogy minél több víz jusson a forrásuktól a ciszternákba. Képes magánál tartani egy part-ot, ami ebben az állapotában nem jelenik meg a játéktéren, és ezt képes elhelyezni adott feltételek teljesülése esetén. Ezen felül képes elemek megjavítására, és a pumpák átállítására.

tároló: olyan dolog, mely számontartja a játékos által kézben tartott elemet.

turn: a játéknak egy olyan időegysége, amely során a soron lévő játékos bármennnyit léphet, de egy interakciójával véget ér (vagy interakció hiányában is, ha jelzi szándékát), és utána a következő játékos turn-je kezdődik el. minden turn elején vízáramlás történik, és a csapatok megkapják a pontjaikat, illetve kis valószínűséggel pumpák romlanak el.

üres szomszéd: egy elem szomszédjának a hiánya az adott irányba, azaz ahol sivatag van

víz: a dolog, ami a források 4 oldalából jön, és a csöveken keresztül áramlik, amíg ki nem folyik egy szabad végű / lyukas csövön, vagy el nem jut egy ciszternába

vízáramlás: az minden turn elején azonnal végbemenő történés, amely során a víz a forrásból egy ciszternába, a csőhálózaton kívülre a sivatagba, vagy egy pumpával elzárt csőbe kerül

2.6 Projekt terv

2.6.1 Használt Technológiák, Eszközök

A projekt megvalósításához használt kulcstechnológiák, a fejlesztést támogató eszközök és programcsomagok.

típus	név
Programozási nyelv	Java (Java 8 SE)
Diagramszerkesztő webapplikáció (GitHub integráció, real-time megoszthatóság)	draw.io/diagrams.net
Verziókezelő rendszer	Git
Forráskódhoszt, projektmenedzsment-platform	GitHub
Időmenedzsment-eszköz	Doodle
Fejlesztői környezet	IntelliJ IDEA
Kollaborációs platform	Discord
Grafikus eszközökészlet	Java Swing
Szövegszerkesztő program	Microsoft Word

2.6.2 A projekt végrehajtásának lépései

A dokumentum „2.1.4 Hivatkozások” szekciójának „Határidők” pontjában szereplő tárgyi honlapon megszabott követelményeket és határidőket vesszük figyelembe a projekt elkészítése során.

A projekt végrehajtásának lépései heti progresszióban történnek, így a fejlesztői csapat által egy előre meghatározott időpontban, minden héten egy meeting keretében kiosztásra kerülnek a feladatok megbeszélés alapján, továbbá a megválasztott kollaborációs platformon (Discord), saját szerveren történnek a kisebb feladat komponensek tervezései, melyhez segítségi szolgálnak a feladat specifikusan létrehozott gondolatmenetek (Threads), illetve hangcsatornák.

2.7 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2023.02.26 14:00	2 óra	Váradi	GitHub repository és szervezet létrehozása, konfigurálása
2023.03.07. 8:00	1 óra	Sasvári Váradi Tepliczky Fekete Mizser	Értekezlet. Döntés: Fekete és Váradi beosztják és elkészítik vasárnap estig: 2.1, 2.2 (kivéve 2-es alpont), 2.6 Sasvári, Tepliczky és Mizser beosztják és elkészítik: 2.2.2, 2.3, 2.4, 2.5 Utólagos értékelés: többnyire tartottuk, de közös megegyezés alapján Discordon folyamatosan konzultálva néhol váltottattunk a feladatkiosztáson
2023.03.07. 11:00	1 óra	Váradi	Dokumentum cél (2.1.1) és Szakterület (2.1.2) megszövegezése
2023.03.07. 11:00	0,5 óra	Fekete	Dokumentum fedőlapjának elkészítése
2023.03.08 17:00	2 óra	Váradi	Dokumentum összefoglalás (2.1.5) megszövegezése, a projekt által használt technológiák meghatározása és dokumentálása (2.6.1)
2023.03.09. 10:00	3 óra	Váradi	Komponens diagram (2.2.1) elkészítése
2023.03.09. 10:00	1 óra	Fekete	Felhasználók (2.2.3) és Korlátozások (2.2.4) megírása
2023.03.11 12:30	1 óra	Váradi	Komponens diagram kiegészítése
2023.03.11. 14:00	4 óra	Mizser	Funkciók és ötletelés (2.2.2) (irányítóként)
2023.03.11. 15:00	3 óra	Tepliczky	Funkciók és ötletelés (segítőként)
2023.03.11. 15:00	3 óra	Sasvári	Funkciók és ötletelés (segítőként)
2023.03.11. 18:00	3 óra	Mizser	Funkcionális köv. (2.3.1) (egyenlő résztvevőként)
2023.03.11. 18:00	3 óra	Tepliczky	Funkcionális köv. (egyenlő résztvevőként)
2023.03.11. 18:30	1 óra	Sasvári	Use-case diagram (2.4.2)
2023.03.11. 19:30	1,5 óra	Sasvári	Funkcionális köv. (egyenlő résztvevőként)
2023.03.11. 22:00	3 óra	Mizser	Szótár (2.5) Erőforrásokkal kapcsolatos követelmények (2.3.2)
2023.03.11. 23:00	1 óra	Tepliczky	Use-case leírások (2.4.1) vázlata
2023.03.12. 4:30	0,5 óra	Sasvári	Use-case diagram javítása
2023.03.12. 5:00	1 óra	Sasvári	Dokumentum szerkesztés
2023.03.12. 6:30	2 óra	Sasvári	Use-case leírások kiegészítése
2023.03.12. 11:00	1 óra	Sasvári	Use-case leírások véglegesítése (egyenlő résztvevőként)
2023.03.12. 11:00	2 óra	Tepliczky	Use-case leírások véglegesítése (egyenlő résztvevőként)
2023.03.12 13:00	0,5 óra	Váradi	Áttekintés (2.2.1) megszövegezése
2023.03.12 15:00	1,5 óra	Váradi	Áttekintés bővítése, és az alábbiak elkészítése: Definíciók (2.1.3), hivatkozások (2.1.4), Átadással kapcsolatos követelmények (2.3.3), Egyéb nem funkcionális követelmények (2.3.4)
2023.03.12. 20:00	0,5 óra	Tepliczky	Dokumentum szerkesztés (egyenlő résztvevőként)
2023.03.12. 20:00	0,5 óra	Sasvári	Dokumentum szerkesztés (egyenlő résztvevőként)
2023.03.12. 20:30	1 óra	Sasvári	Feltételezések, kapcsolatok (2.2.5); A projekt végrehajtásának lépései (2.6.2)
2023.03.12. 21:00	3 óra	Mizser	Funkciók, Funkcionális követelmények, Use-case-ek és a Szótár konziszenciájának ellenőrzése, inkonziszenciák kijavítása.

3. Analízis modell (I. változat)

5 – runtime_error

Konzulens:
Dobos-Kovács Mihály

Csapattagok

Mizser Ádám Zoltán	SHKGZW	mizser.adam@gmail.com
Tepliczky Olivér	WB6LC5	tepliczkyo@edu.bme.hu
Fekete Álmos Valér	KR5WPC	feketealmos0@gmail.com
Váradi Kristóf	BP17IB	kristofvaradi@edu.bme.hu
Sasvári Szabolcs Attila	TWOZG6	szabolcs.attila.sasvari@edu.bme.hu

2023.03.13.

3. Analízis modell kidolgozása

3.1 Objektum katalógus

fontos megjegyzés: A Game (játék) objektumunk a Controller-ünk. A nem játékosokhoz köthető Use-Case-eket ő valósítja meg, illetve ő a híd a View és Model között. Bár az analízis modellnek nem feltétlenül kellene a része lennie, mi mégis megjelenítjük itt, mert számos szekvenciadiagramhoz elengedhetetlen, és ez a tervezést jelentősen megkönnyíti.

3.1.1 Játék

A játékmenetet kezeli (vizsgálja, hogy véget ért-e a játék, valamint a köröket vezérli). Számoltartja és módosítja a csőrendszert (turn-önként véletlenszerűen pumpákat ront el, round-onként pedig ciszternák szomszédos üres helyeire új csöveket helyez le). Turn-ök elején folyatja a vizet a forrásokból a szabad végű / lyukas csövekig vagy ciszternáig, illetve pontokat tud osztani a csapatoknak, amit szintén számoltart.

Irányokat / célpontokat tud bekérni a felhasználótól. Játékosmozgatást, cső- és pumpalehelyezést, illetve csőfelvezést kezdeményezhet vele. Pumpát is tud adni a jelenlegi játékosnak, amennyiben egy szerelő, ciszernán áll és üres a tárolója.

3.1.2 Csőrendszer

Számoltartja a játék elemeit (csöveket, pumpákat, forrásokat és ciszternákat). Ezek közül eltávolíthat csöveket úgy, hogy ne essen komponensekre a csőrendszer, mint gráf. Az eltávolított csövet oda tudja adni annak a szerelőnek, aki az eltávolítást kezdeményezte. A csövek és pumpák elhelyezését is kezeli. Vizsgálja, hogy ezek a műveletek lehetségesek-e egyáltalán.

3.1.3 Forrás

Olyan elem, amely vizet juttat a vele szomszédos csövekbe, azaz a számoltartott kimeneteibe, illetve a sivatagba, ha nincsen szomszédja az adott irányba. Ahányszor előfordul az utóbbi, annyi pontot oszt a szabotőrök csapatának.

Számoltartja a játékosokat, akik rajta állnak. Játékosokat tud felenni, illetve eltávolítani magáról.

3.1.4 Ciszterna

Olyan elem, amely számoltartja a bemeneteit. Amennyi csőből folyik víz az adott ciszternába, annyi pontot oszt a szerelők csapatának.

Számoltartja a játékosokat, akik rajta állnak. Játékosokat tud felenni, illetve eltávolítani magáról.

3.1.5 Pumpa

Olyan elem, amely számoltartja a bemenetét, a kimenetét és a szomszédait. A szomszédai közül be tudja állítani, hogy melyik a bemenete és melyik a kimenete. Vizet képes juttatni a bemeneti csövéből, önmagán keresztül (ő maga is tárol vizet) a kimeneti csövébe. El tud romlani, és ekkor nem képes víz átengedésére. Amikor víz megy át rajta, pontot oszthat a szabotőrök csapatának, ha a kimenete egy szabad végű / lyukas cső vagy sivatag (nincs kimenete).

Számoltartja a játékosokat, akik rajta állnak ezzel együtt játékosokat tud felenni, illetve eltávolítani magáról.

3.1.6 Cső

Olyan elem, amely számontartja a bemenetét és kimenetét, és az előbbiből az utóbbiba vizet képes juttatni. Tárolja a benne lévő vizet. Ki lehet lyukasztani, és meg lehet javítani. Lyukas állapotában nem képes vizet átengedni magán. Amikor vizet enged át, pontot oszthat a szabotőrök csapatának, ha a kimenete egy lyukas cső vagy sivatag (nincs kimenete).

Számontartja a játékost, aki rajta áll, valamint egy játékost tud felenni, illetve eltávolítani magáról (egyszerre egy játékos lehet egy csövön).

3.1.7 Szerelő

Számon tartja azt az elemet, amin éppen áll, valamint szomszédos elemekre tud lépni, hogyha ez lehetséges (azaz akkor, ha a szomszédos csövön esetleg nem áll már másik játékos).

Manipulálni tudja az elemet, amin éppen áll (csövet és pumpát tud javítani, illetve működő pumpát tud átállítani). Át képes lépni szomszédos elemekre. Ha éppen ciszternán áll, lehetősége van átlépni róla egy másikra.

A tárolója tárolhat vagy egy csövet, vagy egy pumpát. El tud távolítani egy szomszédos, üres csövet, hogy ha a tárolója üres, valamint ezt az csövet később elhelyezni egy szomszédos, üres helyre. Pumpát olyan cső helyére tud letenni, aminek a bemenete és a kimenete is egy cső.

3.1.8 Szabotőr

Számon tartja azt az elemet, amin éppen áll, valamint szomszédos elemekre tud lépni, hogyha ez lehetséges (azaz akkor, ha a szomszédos csövön esetleg nem áll már másik játékos).

Manipulálni tudja az elemet, amin éppen áll (csövet tud lyukasztani, illetve pumpát tud átállítani). Át képes lépni szomszédos elemekre. Ha éppen ciszternán áll, lehetősége van átlépni róla egy másikra.

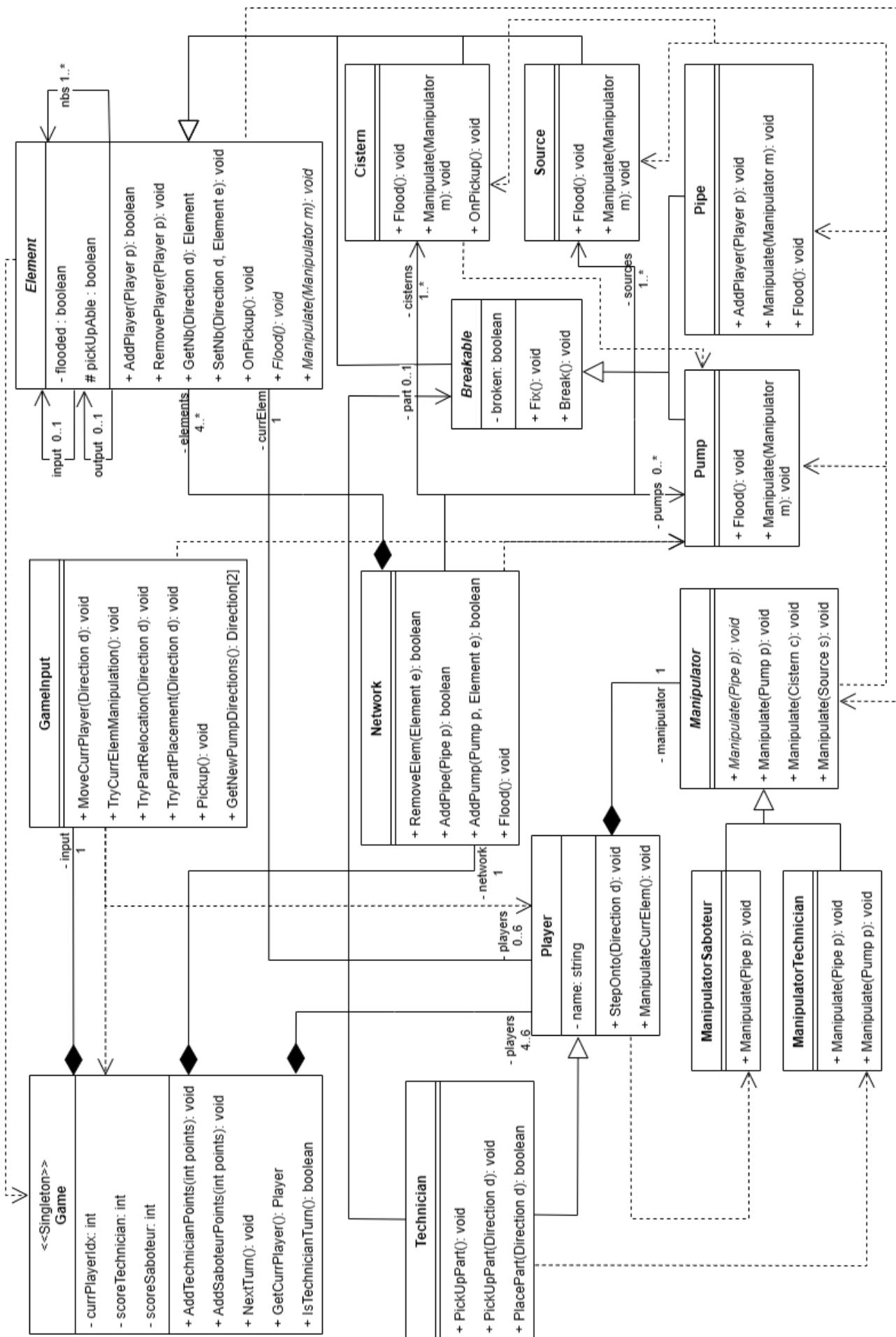
3.2 Statikus struktúra diagramok

fontos megjegyzések:

- A geometriát igyekeztünk eltávolítani. A *Direction* típus a modell szempontjából tetszőleges absztrakció lehet.
- A triviális (paraméter nélküli/egyparaméteres) setter/getter függvényeket, viszont az "Osztályok leírása" szekcióban fel vannak ezek is sorolva.
- Ha egy ōsnek van egy dependenciája, akkor az érvényes minden leszármazottra. A dependenciák csak akkor vannak behúzva, ha nincsen erősebb kapcsolat az osztályok között (leszármazás, kompozíció, aggregáció, asszociáció).

3. Analízis modell kidolgozása

runtime_error



3.3 Osztályok leírása

3.3.1. Breakable

- **Felelősség**

Ez az elem tönkre tud menni/kilyukadni, valamint ez által meg is lehet javítani.

Ilyen elemet tudnak tárolni a szerelők (más néven part).

(Ez egy absztrakt osztály.)

- **Ősosztályok**

Element

- **Attribútumok**

- - **broken:** boolean az adott elem hibás-e vagy sem, azaz át tud ereszteni-e magán vizet.

- **Metódusok**

- + void **Fix()** : Hibás elem megjavítására szolgáló függvény.
- + void **Break()** : Épp elem tönkretevésére szolgáló függvény.
- + void **GetBroken()** : Visszaadja, hogy az adott elem törött-e.
- + boolean **GetPickUpAble()** : Visszaadja, hogy az elem felvehető-e. Az Element űsbéli megvalósításhoz hozzá vesz egy új feltételt: ne legyen törött sem a part, hogy felvehető legyen.

3.3.2. Cistern

- **Felelősség**

Amennyi csőből folyik víz ebbe az elembe, annyi pontot oszt a szerelők csapatának.

Erről az elemről a játékosok közvetlen más ciszternákra tudnak lépni.

- **Ősosztályok**

Element

- **Metódusok**

- + void **OnPickUp()** : megpróbál pumpát adni a soron lévő játékos tárolójába.
- + void **Flood()** : Vízzel tölti fel magát, és pontot ad a szerelőknek. Úgy van felüldefiniálva, hogy ő már nem folyat más elemekbe vizet (végpont).
- + void **Manipulate(Manipulator m)** : az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.3. Element

- **Felelősség**

Player-eket tud felrakni, illetve eltávolítani magáról, és számon is tartja a rajta tartózkodókat. Tárolja a bemenet/kimenetét, valamint a szomszédjait is, ezeket tudja változtatni, illetve lekérdezni. Továbbá tárolja, hogy van-e víz benne.

- **Attribútumok**

- - **flooded:** boolean : Értéke igaz, ha víz van benne (jelenleg vizet tárol).
- # **pickUpAble:** boolean : Értéke igaz, ha az adott elem felvehető, különben hamis.
- - **players:** Player[0..6] : Azon játékosok halmaza, amelyek az adott elemen tartózkodnak
- - **nbs:** Element[1..*] : A vele szomszédos elemekre hivatkozik.
- - **input:** Element[0..1] : A bemenő szomszédra hivatkozik, ha van ilyen.
- - **output:** Element[0..1] : A kimenő szomszédra hivatkozik, ha van ilyen.

• Metódusok

- + void AddPlayer(Player p) : Felhelyez magára egy játékost, ha ez lehetséges a rajta állók száma szerint.
- + void RemovePlayer(Player p) : Eltávolítja a számottartott játékosai közül az átadottat.
- + Element GetNb(Direction d) : Megadott irányba lévő szomszédos elemét adja vissza, ha van ilyen.
- + void SetNb(Direction d, Element e) : Megadott irányú szomszédjának állítja be az átadott elemet.
- + void OnPickup() : Ha egy elem támogatni akarja azt a funkcionálitást, hogy egy szerelő rajta álla egy part-ot kaphasson a tárolójába, akkor ezt a függvényt kell felülírnia. Alapértelmezett megvalósítása üres törzsű függvény (alapjáraton nem támogatják az elemek ezt a funkcionálitást).
- + void Flood() : Absztrakt metódus, amely a származtatott osztályokban felül lesz írva a szerint, hogy hogyan ereszlik át magukon a vizet.
- + void ManipulateManipulator(m) : Absztrakt metódus, mivel az elem konkrét típusától függ a lefolyása. Az átadott manipulátorral manipulálja az elemet.
- + boolean GetPickUpAble() : Visszaadja, hogy az elem felvehető-e („pickUpAble”, nincs benne víz és nem áll rajta egyetlen játékos sem).
- + int GetNbCnt() : A szomszédos elemek számát adja vissza.
- + int GetPlayerCnt() : A rajta tartózkodó játékosok számát adja vissza.
- + boolean GetFlooded() : Igazat ad vissza, ha az víz van benne (vagy víz folyik ki belőle).
- + Element GetInput() : Visszaadja a bemeneti elemét.
- + Element GetOutput() : Visszaadja a kimeneti elemét.
- + void SetFlooded(boolean f) : Az átadott érték szerint állítja be, hogy van-e víz jelenleg benne.
- + void SetInput(Element e) : A bemenetét az átadott elemre állítja be.
- + void SetOutput(Element e) : A kimenetét az átadott elemre állítja be.

3.3.4. Game

• Felelősség

A játékmenetet kezeli (vizsgálja, hogy véget ért-e a játék, valamint a köröket vezérli). Számottartja és módosítja a csőrendszert (turn-önként véletlenszerűen pumpákat ront el, round-önként pedig ciszternák szomszédos üres helyeire új csöveket helyez le). Turn-ök elején folytatja a vizet a forrásokból a szabad végű / lyukas csövekig vagy ciszternáig, illetve pontokat tud osztani a csapatoknak, amit szintén számottart.

• Attribútumok

- - currPlayerIdx: int : A soron lévő játékos indexe a „players” gyűjteményben.
- - scoreTechnician: int : A szerelők pontszámát tárolja.
- - scoreSaboteur: int : A szabotőrök pontszámát tárolja.
- - input: Gameinput : A felhasználóval kommunikáló objektum.(Híd a View és a Model között.)
- - players: Player[4..6] : A játékban résztvevő játékosokat tárolja.
- - network: Network : A pálya elemeit tárolja, szortírozza.

• Metódusok

- + void AddTechnicianPoints(int points) : Ez a metódus ad a szerelőknek pontot.
- + void AddSaboteurPoints(int points) : Ez a metódus ad a szabotőröknek pontot.
- + void NextTurn() : A következő turn indítása (frissíti/folytatja a vizet, pontokat oszt, véletlenszerűen pompákat ront el, és ha lement egy teljes kör (round), akkor új csöveket teremt a ciszternák üres szomszédjai helyén). Ha az egyik csapat elérte a győzelemhez szükséges pontok számát, véget vet a játéknak.
- + Player GetCurrPlayer() : Visszaadja a játékost, aki éppen soron van (akinek turn-je van jelenleg).
- + boolean IsTechnicianTurn() : Igazat ad vissza, ha éppen szerelő jön az adott körben.
- + GameInput GetInput() : Visszaadja a felhasználóval kommunikáló objektumot.
- + Network GetNetwork : Visszaadja a pálya elemeit számottartó objektumot.

3.3.5. GameInput

- **Felelősség**

A felhasználót és a játékot köti össze. Irányokat / célpontokat tud bekérni a felhasználótól. Játékosmozgatást, cső- és pumpalehelyezést, illetve csőfelvezést kezdeményezhet vele. Pumpát is tud adni a jelenlegi játékosnak, amennyiben egy szerelő, ciszernán áll és üres a tárolója.

- **Metódusok**

- + void MoveCurrPlayer(Direction d) : d irányba próbálja mozgatni az épp soron lévő játékos, arról az elemről, amelyiken éppen tartózkodik.
- + void TryCurrElemManipulation() : akkor hívandó függvény, amikor éppen azt az elemet szeretné manipulálni / interakcióba lépni vele (csövet lyukasztani / foltozni, pumpát javítani / átállítani vagy a következő ciszternára ugrani) a soron lévő játékos, amelyiken éppen áll.
- + void TryPartRelocation(Direction d) : akkor hívandó függvény, amikor a soron lévő játékos megpróbál egy tőle d irányba lévő szomszédos csövet felvenni, és a tárolójában eltárolni.
- + void TryPartPlacement(Direction d) : akkor hívandó függvény, amikor a soron lévő játékos megpróbálja a tárolt part-ját tőle d irányba lehelyezni.
- + void Pickup() : akkor hívandó függvény, amikor egy játékos egy part-ot próbálva a tárolójába tenni (nem a játéktérrel, hanem közvetlenül odaadva). Ez csak akkor lesz sikeres, ha a játékos szerelő, üres a tárolója, és az elem amin áll, támogat ilyen funkcionálitást.
- + Direction[2] GetNewPumpDirections() : pumpa átállításához visszaadja a soron lévő játékos által bevitt irányokat.

3.3.6. Manipulator

- **Felelősség**

Absztrakt osztály, amely a Visitor tervezési mintát valósítja meg a leszármazottaival együtt. A leszármazottai egy játékos típusnak készítik el a „gazda” elem manipulálását megvalósító viselkedést minden elem esetére (pumpák, csövek, források, ciszternák). A szerelők konkrét manipulátora például definiálja, hogy mit tud tenni egy szerelő, ha az előbb említett elemeken állva próbál interaktálni. A függvényei gyakran a kört is léptetik.

- **Metódusok**

- + void Manipulate(Pipe p) : Absztrakt metódus a játékosok cső manipulálásának leírására.
- + void Manipulate(Pump p) : Átállítja az átadott pumpát (GameInput-ot használja a bemenetért). Ezután véget ér a játékos köre (turn).
- + void Manipulate(Cistern c) : Átlépteti a jelenlegi játékos a következő ciszternára. Ezzel nem ér véget a játékos köre (turn).
- + void Manipulate(Source s) : Üres törzsű függvény, jelenleg a játékosok nem tesznek semmit a forráson állva. (A jövőbeli bővíthetőségre fenntartva, illetve a Dynamic Dispatch hibátlan működése miatt kell, hogy ne kelljen Type-checkingelni a hívóoldalon.)

3.3.7. ManipulatorSaboteur

- **Felelősség**

A szabotőrök konkrét manipulátora, ami definiálja, hogy hogyan manipulálhatják az összes lehetséges „gazda” elemüket.

A csövek kilyukasztásának viselkedését valósítja meg. A többi viselkedés megfelel az ōsbélivel.

- **Ősosztályok**

Manipulator

- **Metódusok**

- + void **Manipulate(Pipe p)** : Kilyukasztja az átadott p csövet. Utána pedig véget ér a jelenlegi játékos köre.

3.3.8. ManipulatorTechnician

- **Felelősség**

A szerelők konkrét manipulátora, ami definiálja, hogy hogyan manipulálhatják az összes lehetséges „gazda” elemüket.

A csövek javításának viselkedését valósítja meg, illetve a pumpák megjavításának viselkedésével helyettesíti a pumpaátállítást, ha az adott pumpa rossz.

A többi viselkedés megfelel az űsbélivel.

- **Ősosztályok**

Manipulator

- **Metódusok**

- + void **Manipulate(Pipe p)** : Megjavítja az átadott p csövet. Utána pedig véget ér a jelenlegi játékos köre.
- + void **Manipulate(Pump p)** : Megjavítja az átadott p pumpát, ha az elromlott, különben pedig átállítja (ez esetben meghívja az űsbéli megvalósítását a függvénynek).

3.3.9. Network

- **Felelősség**

Tárolja a pálya elemeit. Csövek és pumpák adhatók hozzá, a csöveket pedig el is lehet távolítani róla. Vizsgálja, hogy ezek a műveletek lehetségesek-e egyáltalán.

A benne tárolt elemek azok, amik ténylegesen meg is lesznek jelenítve a pályán.

Továbbá ez az osztály indítja el a vízfolyást forrásokból.

- **Attribútumok**

- - elements: Element[4..*] : A pálya összes elemét tárolja.
- - cisterns: Cistern[1..*] : Külön csoportosítja minden elem közül a ciszternákat ez a gyűjtemény.
- - sources: Source[1..*] : Külön csoportosítja minden elem közül a forrásokat ez a gyűjtemény.
- - pumps: Pump[0..*] : Külön csoportosítja minden elem közül a pumpákat ez a gyűjtemény.

- **Metódusok**

- + boolean **RemoveElem(Element e)** : Megkísérli eltávolítani az átadott elemet a pályáról. Ha ennek semmi akadálya nem volt (pl. nem esne komponensekre a pálya, mint gráf), akkor igazzal tér vissza. Ellenkező esetben nem módosít semmi a pályán.
- + boolean **AddPipe(Pipe p)** : Az átadott csövet megkísérli hozzáadni a pályához. Ennek sikerességét adjá vissza.
- + boolean **AddPump(Pump p, Element e)** : Az átadott pumpával kísérli meg felülríni a másik átadott elemet a pályán. Ennek sikerességét adjá vissza.
- + void **Flood()** : Felapasztja az összes vizet a pályáról, majd minden forrásból elindítja a vizet, aminek következtében el lesz árasztva vízzel a pálya, és pontokat fognak kapni a csapatok.
- + Element[4..*] **GetElements()** : Visszaadja a pálya minden elemét.
- + Pump[0..*] **GetPumps()** : Visszaadja a pályán lévő pumpákat.
- + Cistern[1..*] **GetCisterns()** : Visszaadja a pályán lévő ciszternákat.

3.3.10. Pipe

- **Felelősség**

Olyan elromolható, felvehető elem, amelyen legfeljebb egy játékos tartózkodhat.

Ki lehet lyukasztani, a lyukas csövet pedig meg is lehet javítani. Ha a cső nem lyukas akkor a kimenetéhez vizet tud juttatni.

Ha lyukas, vagy a kimenete üres, akkor a szabotörök pontot kapnak, amikor víz érkezik belé.

- **Ősosztályok**

Element → Breakable

- **Metódusok**

- + void AddPlayer(Player p) : Felhelyezi az átadott játékost magára, ha nem áll rajta már más valaki.
- + void Manipulate(Manipulator m) : az átvett manipulátorral manipulálja ezt a konkrét elemet.
- + void Flood() : Vízzel tölti meg magát. Ha lyukas a cső vagy nincs output-ja akkor pontot kapnak a szabotörök, különben pedig hívja tovább az output-ján a Flood() függvényt (ereszti tovább a vizet).

3.3.11. Player (/ Saboteur)

- **Felelősség**

A Player osztály a szabotőr osztállyal ekvivalens, tehát a szabotörök csapatának játékosai Player típusúak. Eltárolja a játékos nevét, az elemet, amelyen éppen áll, és a manipulátorát. Tud mozogni irányokba, és manipulálni tudja a „gazda” elemét.

- **Attribútumok**

- - currElem: Element : Az adott elem, amelyen a játékos tartózkodik
- - name: string : A játékos neve, azonosítója
- - manipulator: Manipulator : A játékos által használt manipulator objektum a „gazda” elemmel való interakciók lekezeléséhez.

- **Metódusok**

- + void MoveTo(Element e) : Az átadott elemre helyezi a játékost, ha ez lehetséges.
- + void StepOnto(Direction d) : A játékost a megadott irányban lévő elemre lépteti, ha ez lehetséges.
- + void ManipulateCurrElem() : Manipulálja azt az elemet, amelyen éppen tartózkodik. A manipulátora határozza meg, hogy mely „gazda” elem típus esetén mit tesz.

3.3.12. Pump

- **Felelősség**

Olyan elromolható, felvehető elem, amely vizet képes átereszteni a bemenetként beállított csövéről a kimenetként beállított csövére. Pontot oszt a szabotöröknek, ha nincsen kimenete, amikor vizet ereszte át.

- **Ősosztályok**

Element → Breakable

- **Metódusok**

- + void Flood() : Ha a pumpa el van romolva, nem tesz semmit. Ha nincs elromolva akkor vízzel tölti fel magát. Ha nincsen outputja, akkor a szabotörök pontot kapnak, de ha van outputja, akkor az outputra hívja tovább a Flood() függvényt.
- + void Manipulate(Manipulator m) : az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.13. Source

- **Felelősség**

Olyan elem, amely vizet juttat a vele szomszédos elemekbe, és soha sincsen bemenete.

- **Ősosztályok**

Element

- **Metódusok**

- + void **Flood()** : Vízzel tölti fel magát, és minden nem lyukas szomszédjára hívja tovább a Flood() függvényt, azaz ereszti tovább a vizet. Ha nem tudja ezt megtenni az adott szomszédja felé, mert lyukas, vagy nincs arra szomszédja, akkor pontot ad a szabotőröknek.
- + void **Manipulate(Manipulator m)** : az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.14. Technician

- **Felelősség**

Olyan játékos, aki csövek lyukaszítása helyett javítani tudja őket, illetve pumpákat is. Emellett képes vagy egy csövet, vagy egy pumpát tárolni magánál, ami el is tud helyezni a megfelelő feltételek fennállása esetén.

- **Ősosztályok**

Player

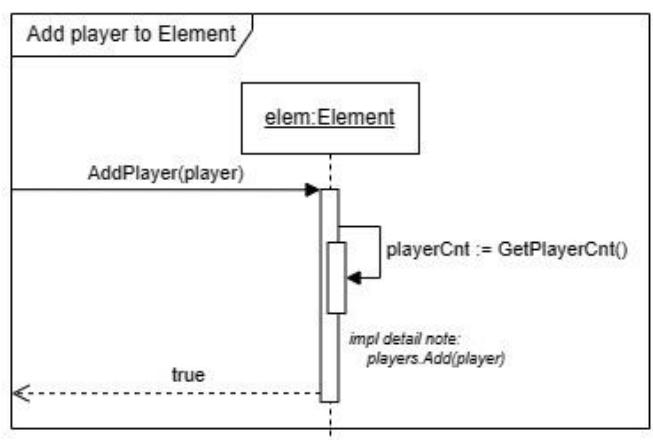
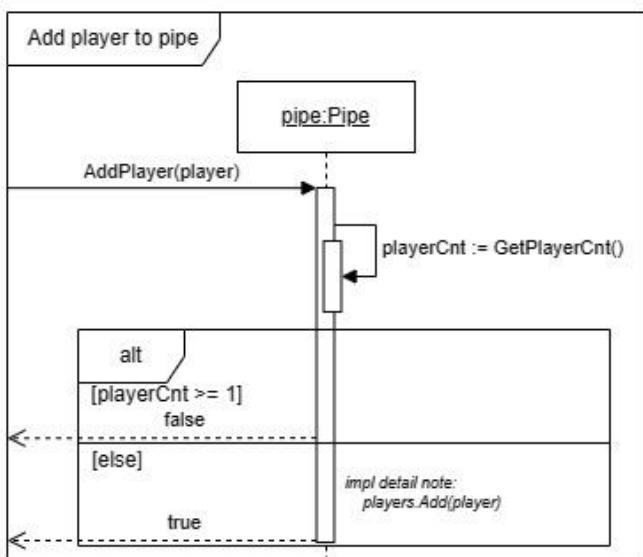
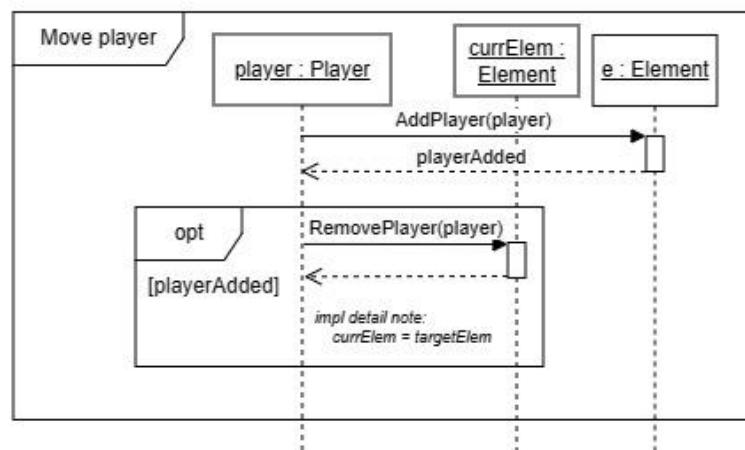
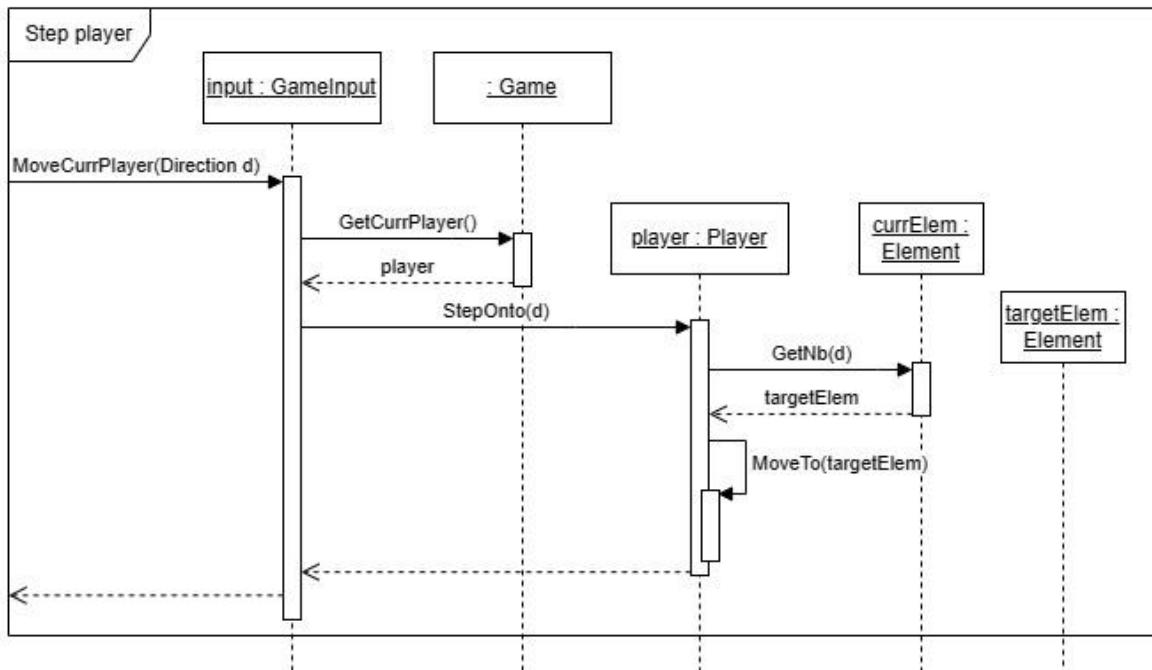
- **Attribútumok**

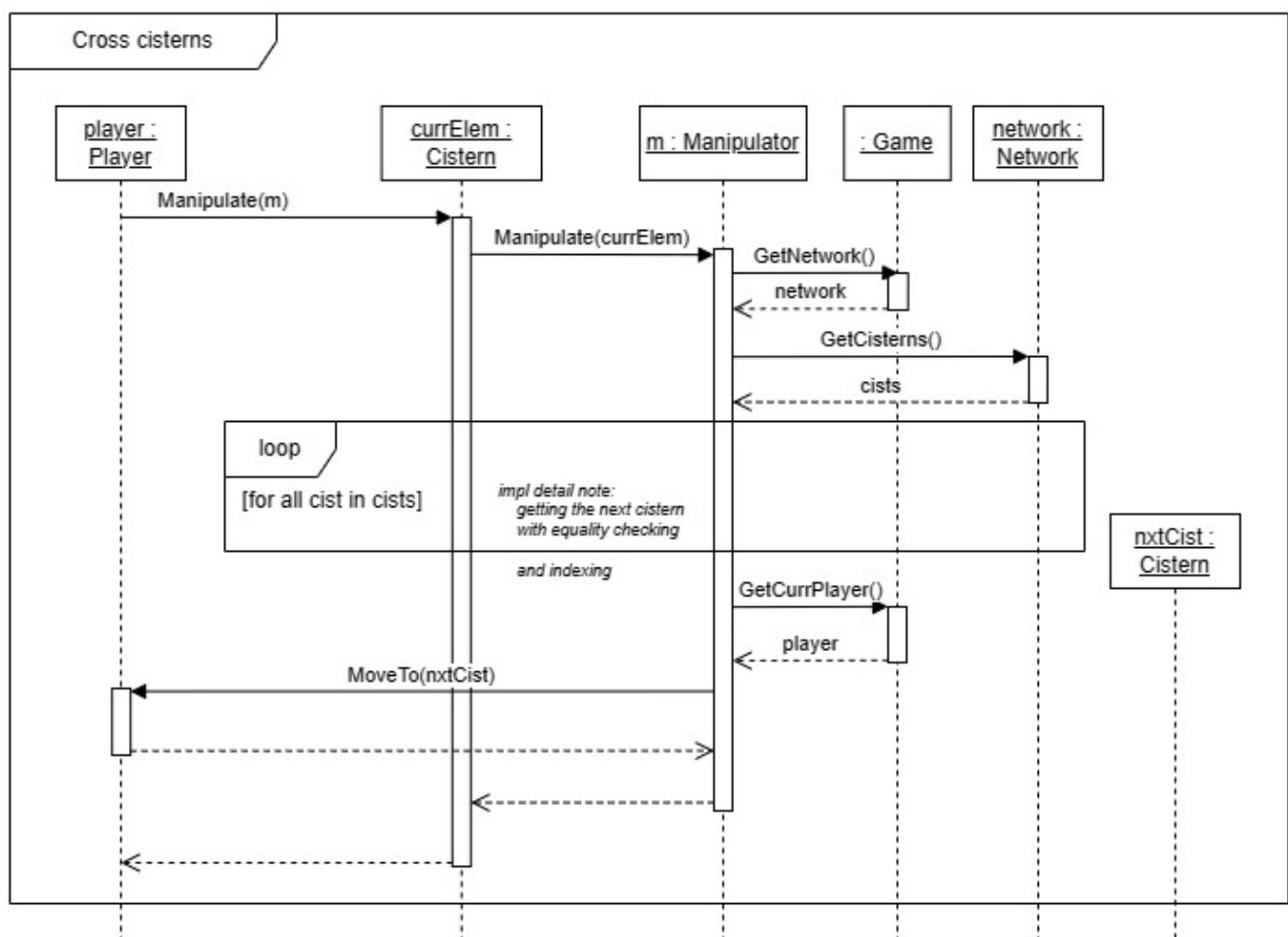
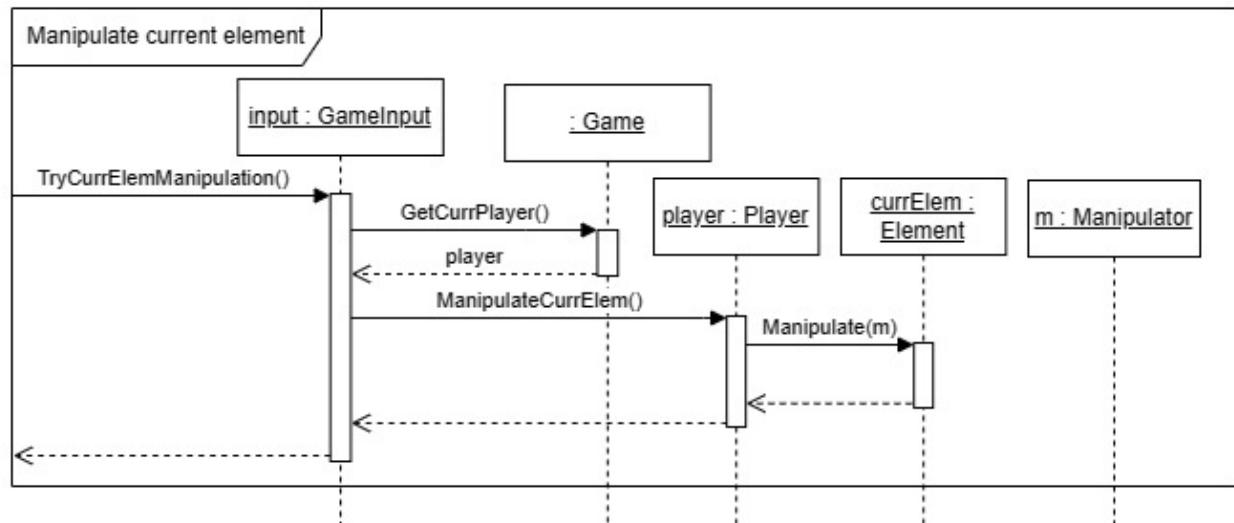
- - part: **Breakable[0..1]** : Az az elem, amelyet a szerelő felvett a pályáról.

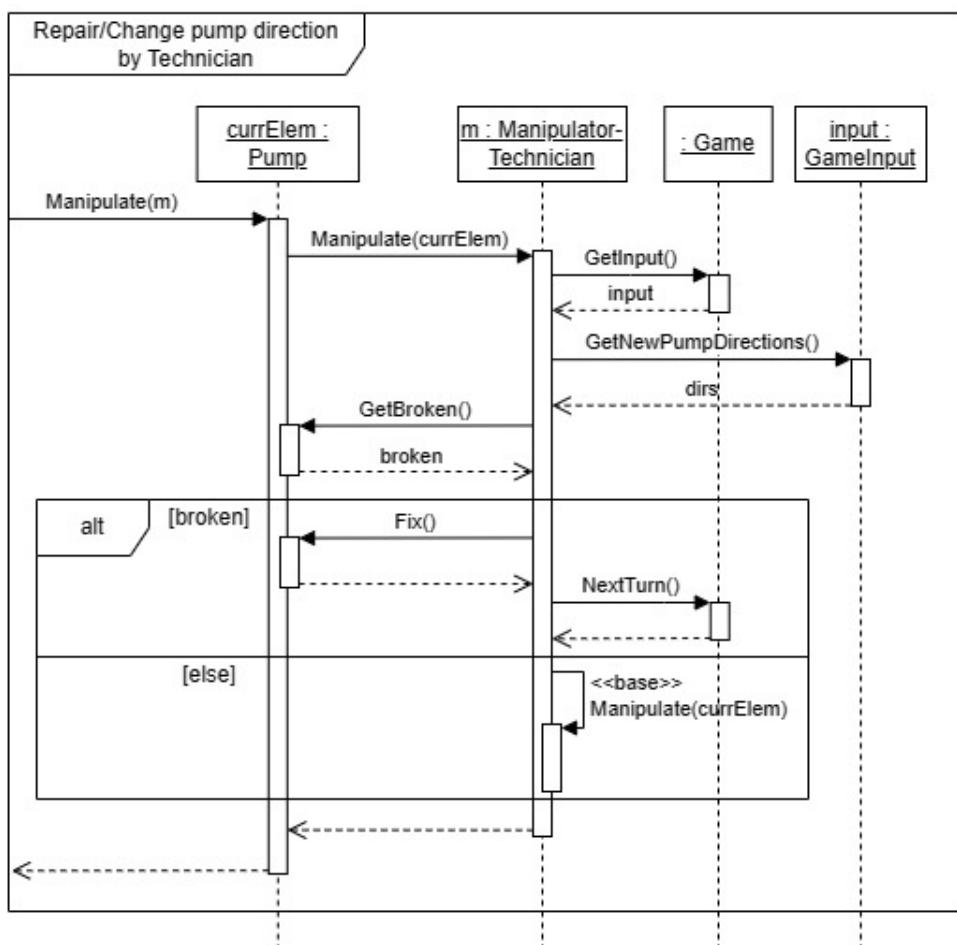
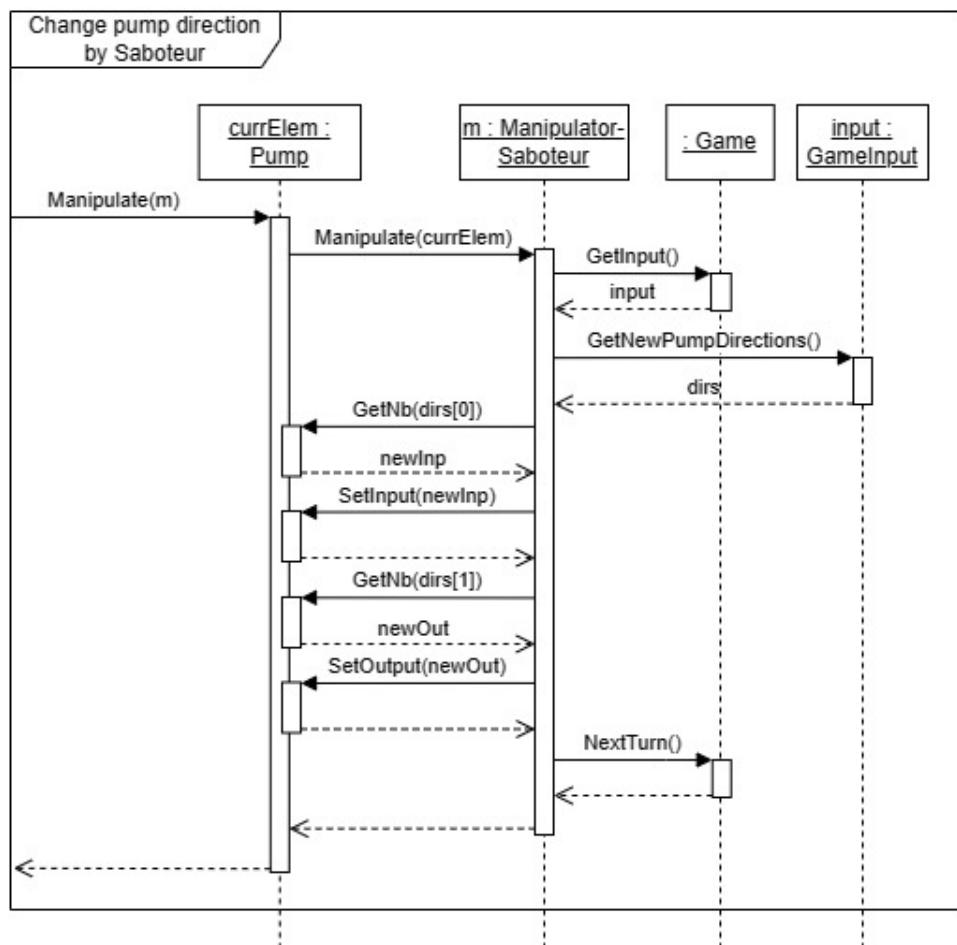
- **Metódusok**

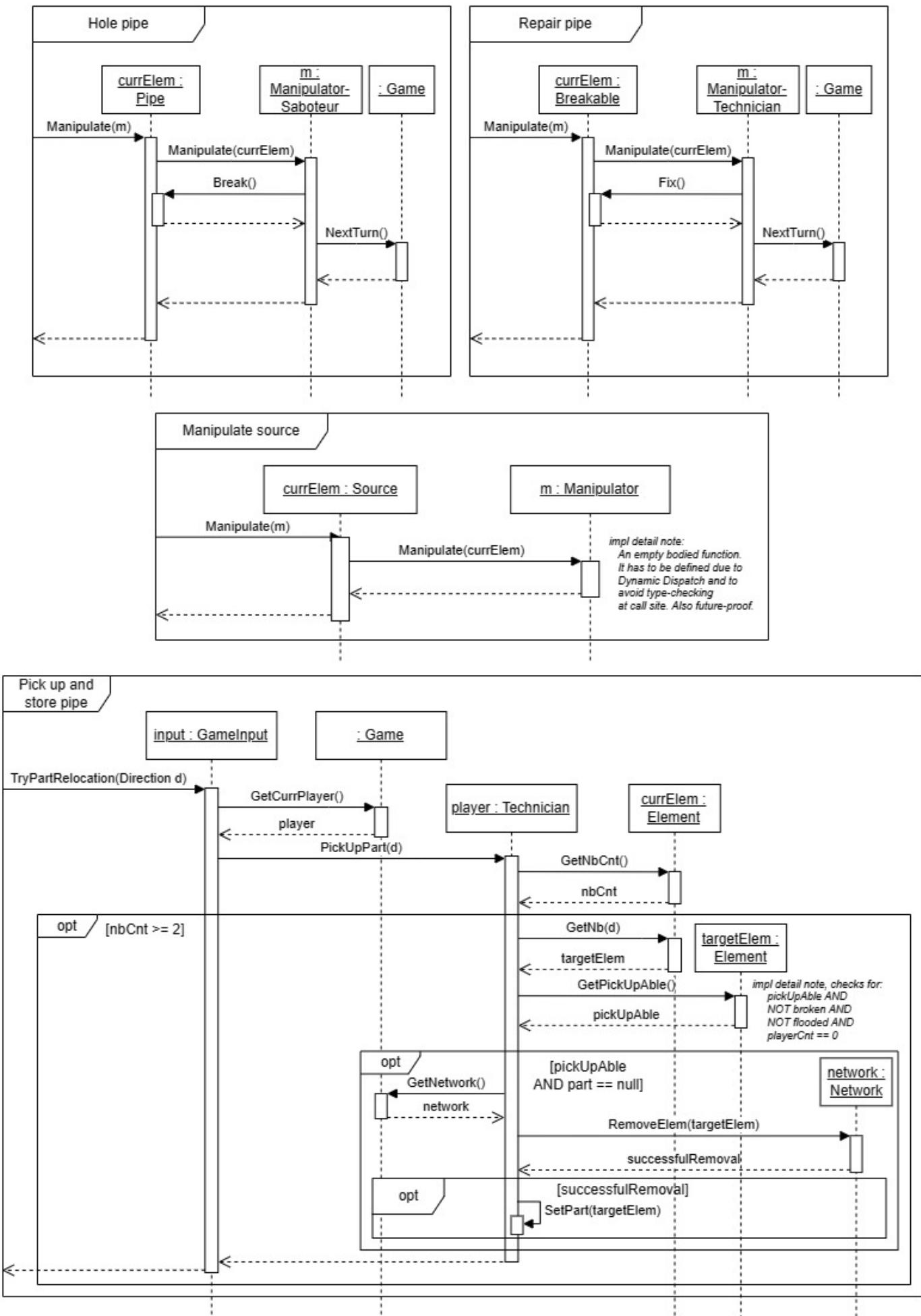
- + void **PickUpPart()** : Megkísérel felvenni egy part-ot a jelenlegi elemtől.
- + void **PickUpPart(Direction d)** : Megkísérli felvenni a d irányban lévő part-ot.
- + boolean **PlacePart(Direction d)** : Megkíséri elhelyezni a tárolt part-ját d irányba. A művelet sikereségével tér vissza.
- + **Breakable GetPart()** : Visszaadja azt a Breakable-t („part” attribútumot), ami a szerelőnél van.
- + void **SetPart(Breakable b)** : Az átadott Breakable-re állítja a „part” attribútumot.

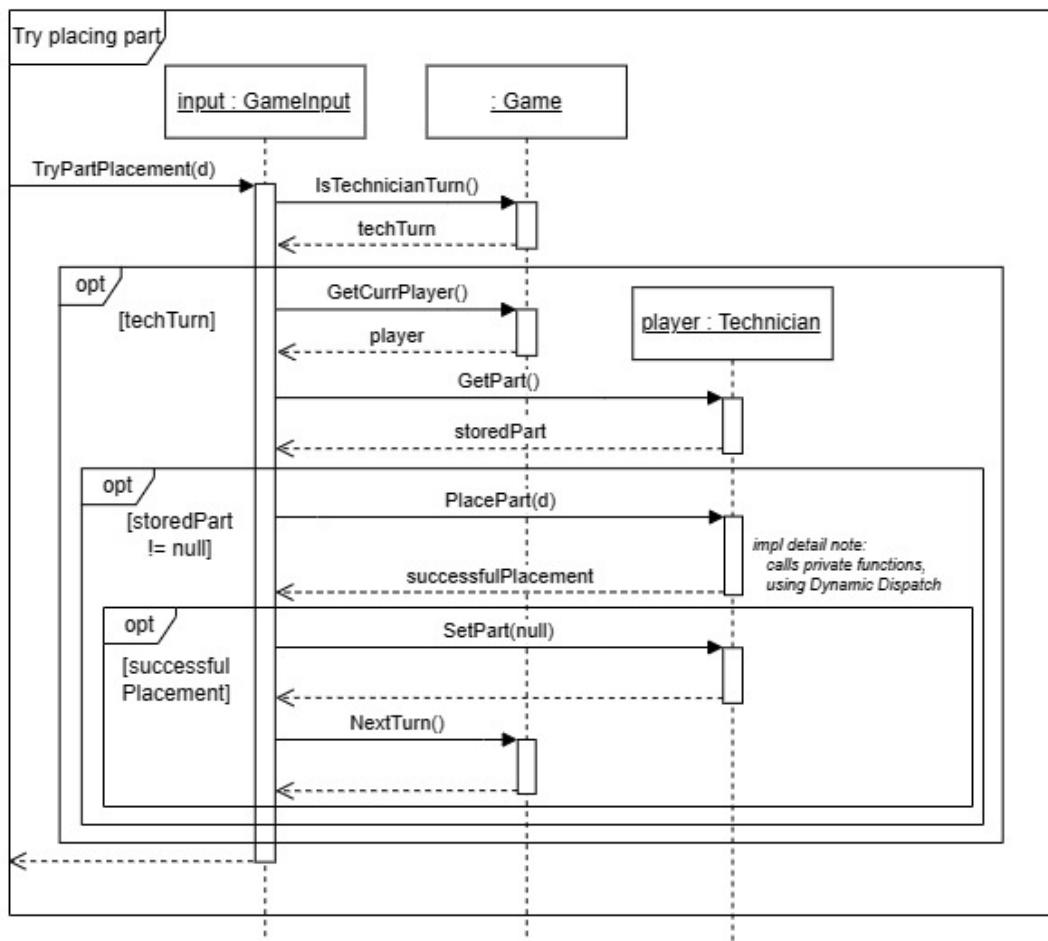
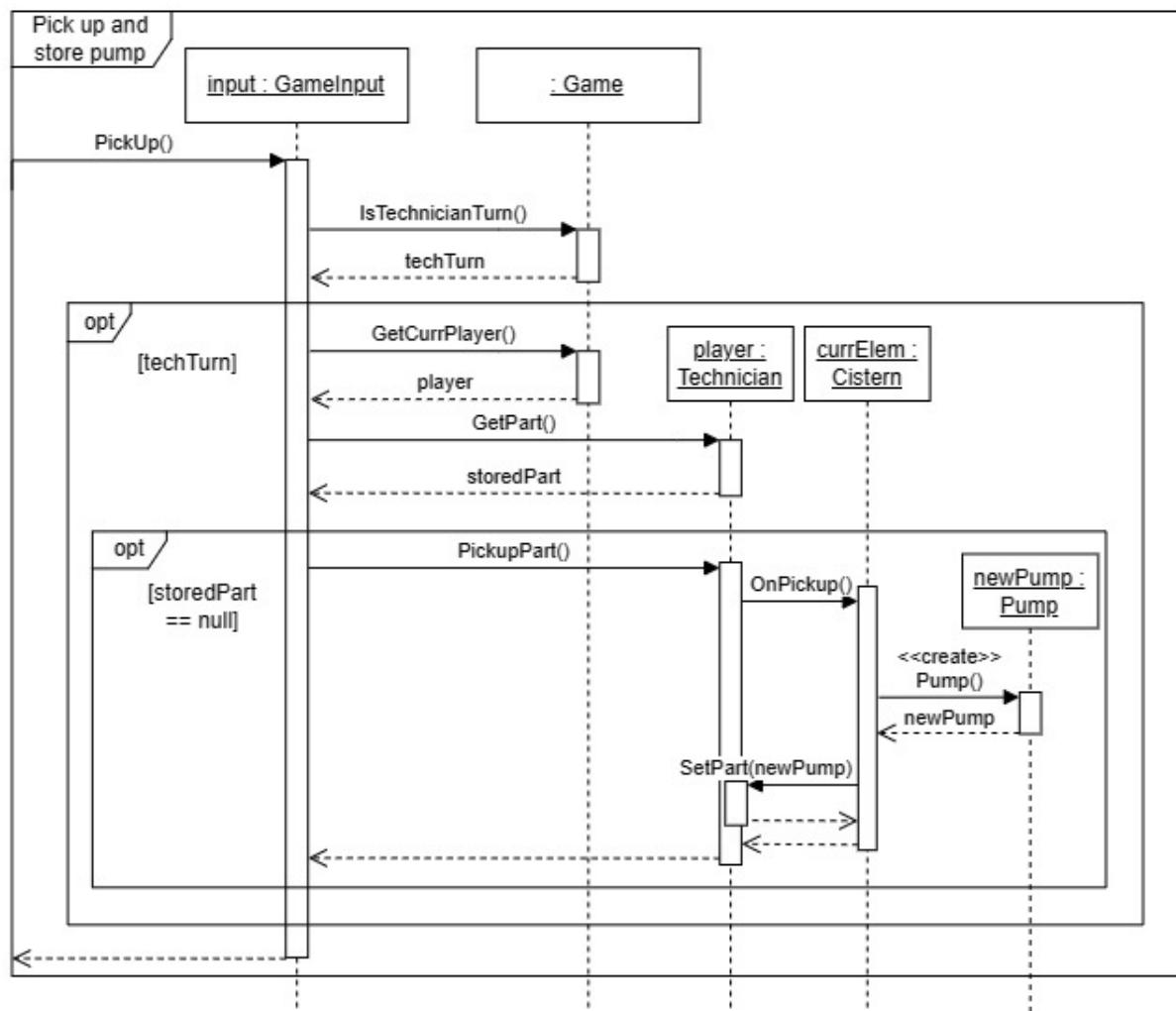
3.4. Szekvencia diagramok

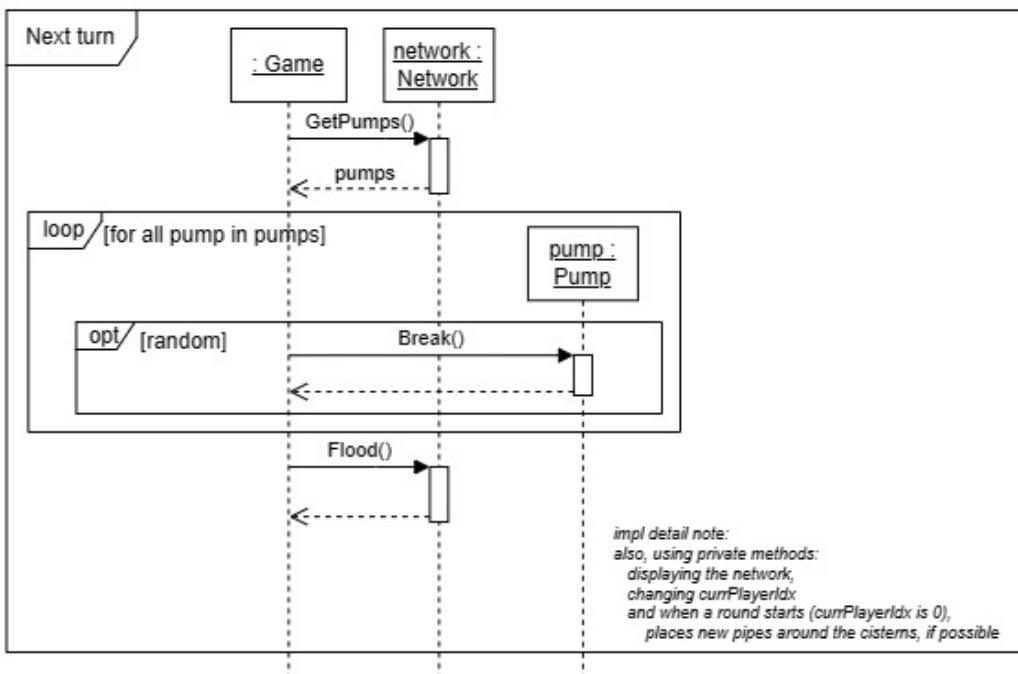
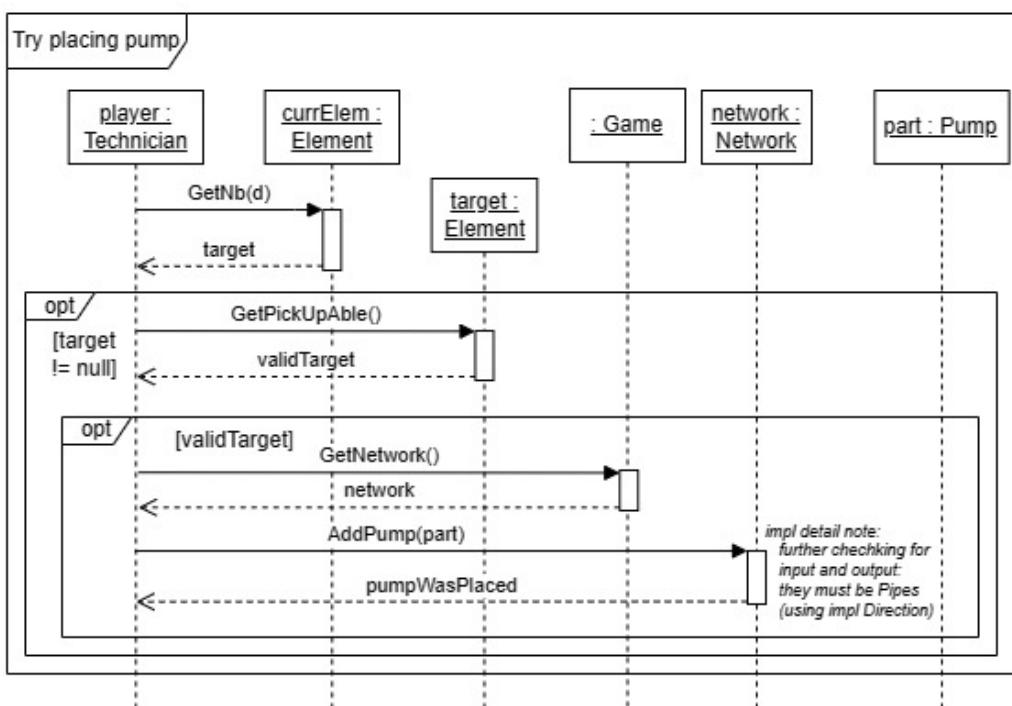
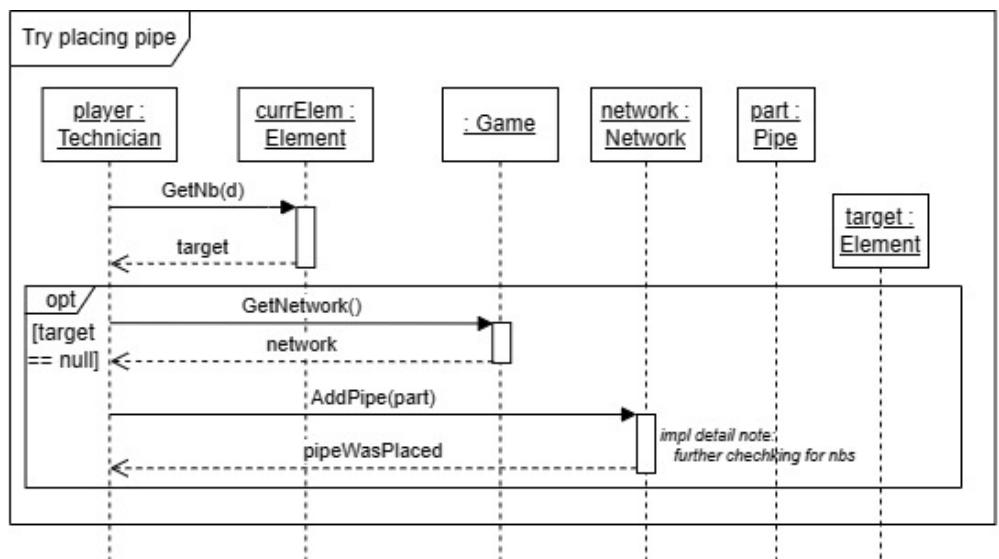


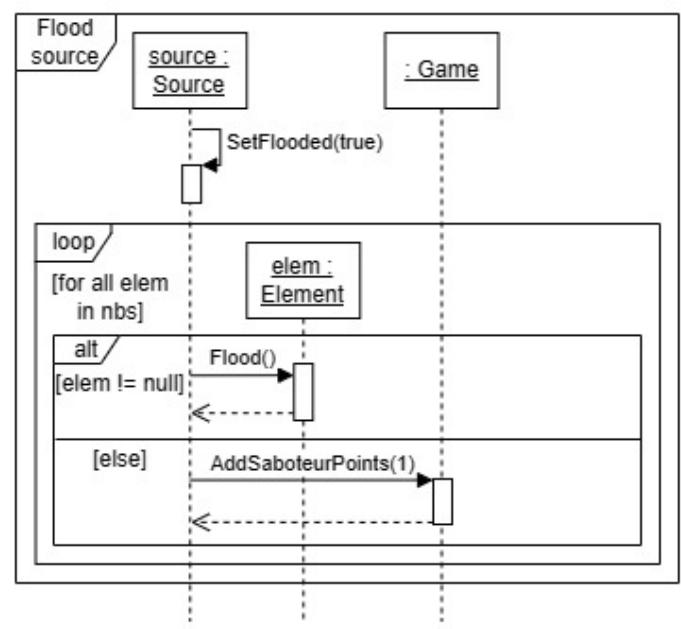
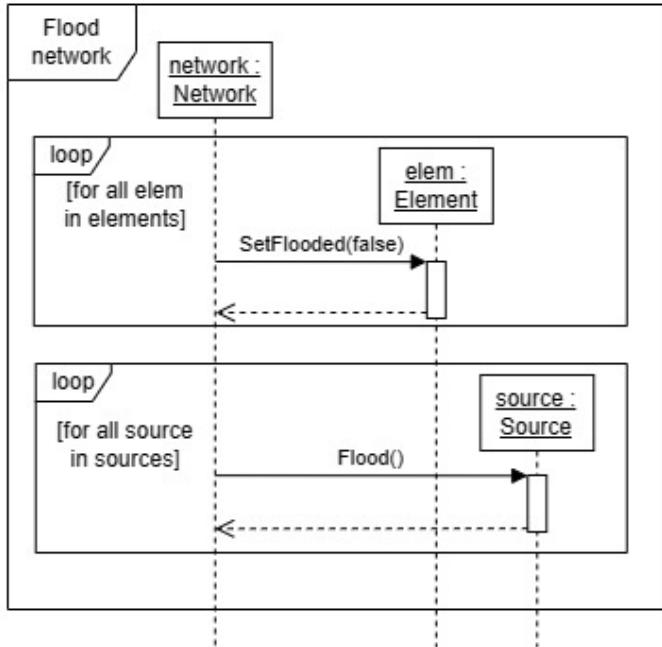




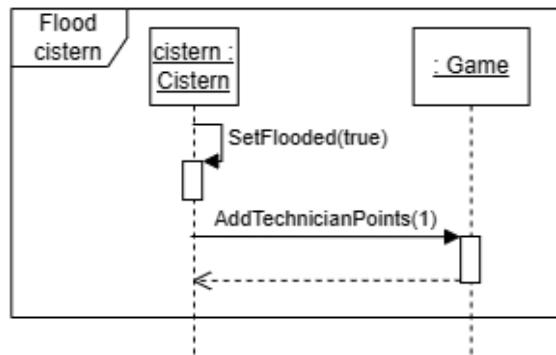
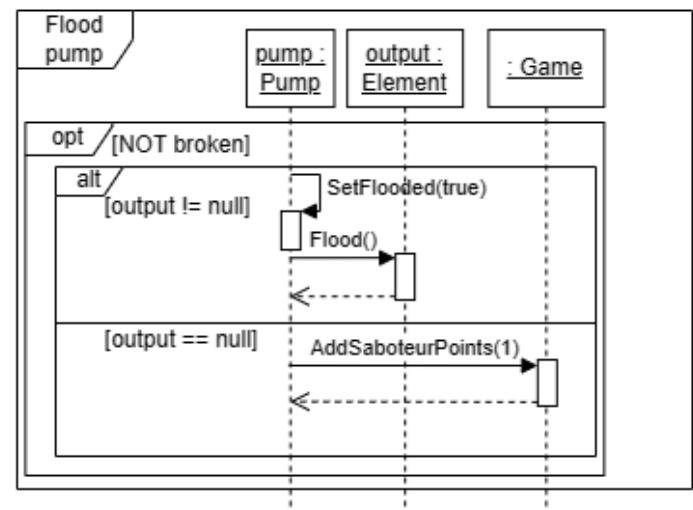
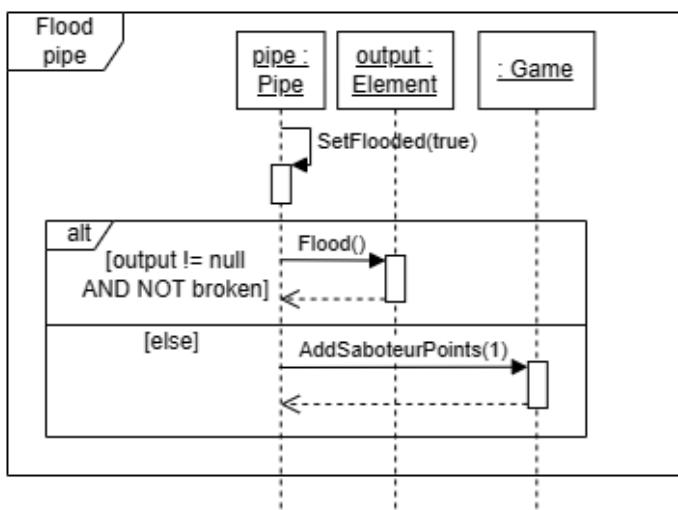






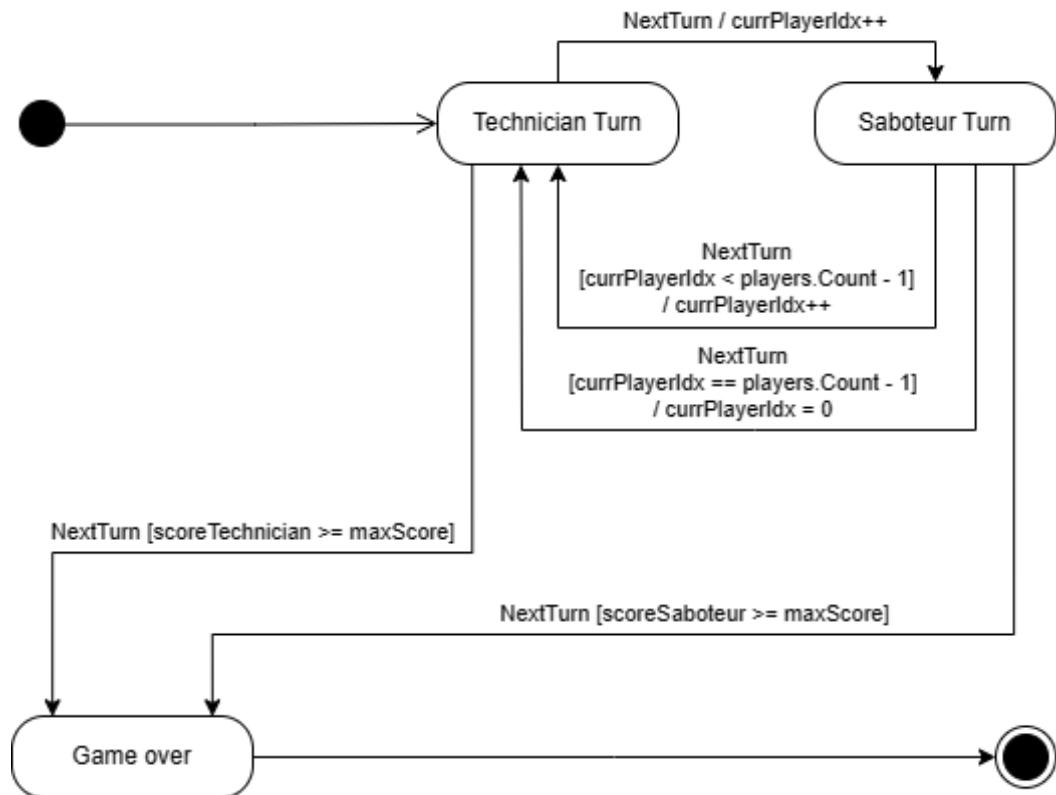


*impl detail note:
"nbs" and "output" are available through Getters*



3.5. State-chartok

3.5.1 Játékosok köreinek (turn) váltakozása



3.6. Napló

Kezdet	Időtartam	Résznevők	Leírás
2023.03.15. 16:30	0,5 óra	Mizser Váradi Tepliczky Fekete Sasvári	<p>Értekezlet. Döntés: Az eheti feladatokat kollaboratív módon, specifikus felosztás nélkül végezzük.</p> <hr/> <p>Objektum katalógus Határidő: 2023.03.16. 8:00. Elkészült: 2023.03.15. 19:00</p> <p>Osztálydiagram, Osztályok leírása Határidő: 2023.03.20. 6:00 Elkészült: 2023.03.20. 0:00</p> <p>Szekvencia diagramok Határidő: 2023.03.20. 11:00 Elkészült: 2023.03.20. 1:00</p>
2023.03.15. 17:00	2 óra	Mizser Váradi Tepliczky Fekete Sasvári	Objektum katalógus meghatározása
2023.03.16 20:00	2,5 óra	Mizser Tepliczky Fekete	Osztálydiagram vázlata
2023.03.16 20:30	2 óra	Sasvári	Osztálydiagram vázlata
2023.03.17. 6:00	1,5 óra	Váradi	Osztálydiagram szerkesztése
2023.03.17 22:00	1 óra	Fekete	Osztályok leírásának szerkesztése
2023.03.18 14:00	1 óra	Mizser Tepliczky Fekete Váradi	Osztálydiagram szerkesztése
2023.03.18 15:00	2 óra	Tepliczky Fekete	Objektum katalógus szerkesztése
2023.03.18 15:00	2 óra	Mizser	Osztálydiagram további részének megalapozása
2023.03.18 17:00	1 óra	Mizser Tepliczky Fekete	Osztálydiagram bővítése
2020.03.18. 21:00	2 óra	Mizser	Visitor minta megtervezése a Manipulator osztályhoz
2023.03.19. 11:30	1,5 óra	Sasvári	Objektum katalógus, Osztálydiagram Review

2023.03.19. 14:00	1 óra	Sasvári	Osztálydiagram bővítése
2023.03.19. 14:00	2 óra	Mizser	Osztálydiagram bővítése
2023.03.19. 15:00	4 óra	Váradi	Osztálydiagram bővítése
2023.03.19. 15:30	5 óra	Fekete	Osztályok leírásának bővítése
2023.03.19. 15:45	4,75 óra	Tepliczky	Osztályok leírásának bővítése
2023.03.19. 16:00	4,5 óra	Sasvári	State-chartok, Szekvencia diagramok
2023.03.19. 21:00	4 óra	Mizser Fekete	Szekvencia diagramok befejezése, osztálydiagram- és -leiársok módosítása

M: 17

V: 13.5

T: 13.75

F: 19

S: 13

3. Analízis modell (II. változat)

5 – runtime_error

Konzulens:
Dobos-Kovács Mihály

Csapattagok

Mizser Ádám Zoltán	SHKGZW	mizser.adam@gmail.com
Tepliczky Olivér	WB6LC5	tepliczkyo@edu.bme.hu
Fekete Álmos Valér	KR5WPC	feketealmos0@gmail.com
Váradi Kristóf	BP17IB	kristofvaradi@edu.bme.hu
Sasvári Szabolcs Attila	TWOZG6	szabolcs.attila.sasvari@edu.bme.hu

2023.03.13.

Analízis modell kidolgozása

3.1 Objektum katalógus

fontos megjegyzés: A Game (játék) objektum a játékvezérlő kontrollert valósítja meg. A nem játékosokhoz köthető használati esetekért felel, illetve ez a híd a View és Model között.

Bár nem képezi az analízis modell részét, meg kell jelenítenünk, mert számos szekvenciadiagramhoz elengedhetetlen, valamint ez a tervezést is jelentősen megkönnyíti.

3.1.1 Játék

A játékmenetet kezeli (vizsgálja, hogy véget ért-e a játék, valamint a köröket vezérli). Számítja és módosítja a csőrendszer felépítését (turn-önként véletlenszerűen pumpákat ront el, round-onként pedig ciszternákkal szomszédos üres helyekre új csöveket helyez). A turn-ök elején a forrásokból áramoltatja a vizet a csövekig, vagy ciszternáig. Pontokat oszt a csapatoknak, és számon is tartja azokat.

Irányokat / célpontokat tud bekérni a felhasználótól. Játékosmozgatást, cső- és pumpalehelyezést, illetve csőfelvételt kezdeményezhet vele. Pumpát adhat a ciszternán található szerelőnek, ha üres a tárolója.

3.1.2 Csőrendszer

3.1.3 Forrás

Olyan elem, amely vizet juttat a vele szomszédos csövekbe, azaz a számított kimeneteibe, illetve a sivatagba, ha nincsen szomszédja az adott irányban. Ahányszor előfordul az utóbbi, annyi pontot oszt a szabotőrök csapatának.

Játékosokat tárol és távolít el az adott pozíóról.

3.1.4 Ciszterna

Olyan elem, amely számítja a bemeneteit. Amennyi csőből folyik víz az adott ciszternába, annyi pontot oszt a szerelők csapatának.

Számítja a játékosokat, akik rajta állnak. Játékosokat tud felenni, illetve eltávolítani magáról.

3.1.5 Pumpa

Olyan elem, amely számítja a bemenetét, a kimenetét és a szomszédait. A szomszédai közül be tudja állítani, hogy melyik a bemenet és melyik a kimenet. Vizet képes juttatni a bemeneti csövéből, önmagán keresztül (ő maga is tárol vizet) a kimeneti csövébe. El tud romlani, és ekkor nem képes víz átengedésére. Amikor víz megy át rajta, pontot oszthat a szabotőrök csapatának, ha a kimenete egy szabad végű / lyukas cső vagy sivatag (nincs kimenete).

Számítja a játékosokat, akik rajta állnak ezzel együtt játékosokat tud felenni, illetve eltávolítani magáról.

3.1.6 Cső

Olyan elem, amely számítja a bemenetet és kimenetet, és az előbbiből az utóbbiba vizet képes juttatni. Tárolja a benne lévő vizet. Ki lehet lyukasztani, és meg is lehet javítani. Lyukas állapotában nem képes vizet átengedni magán. A víz átfolyásakor pontot oszt a szabotöröknek, amennyiben a cső lyukas, vagy kimenete a sivatag.

Számítja a játékost, aki rajta áll, valamint egy játékost tud felenni, illetve eltávolítani magáról (egyszerre egy játékos lehet egy csövön).

3.1.7 Szerelő

Számítja azt az elemet, amin éppen áll. Szomszédos elemre léphet, hogyha ez lehetséges (aza, ha a szomszédos csövön esetleg nem áll másik játékos). A cisternák között direkt módon mozoghat.

Manipulálni tudja az elemet, amin éppen áll (csövet és pumpát tud javítani, illetve működő pumpát tud átállítani).

A tárolója tárolhat vagy egy csövet, vagy egy pumpát. El tud távolítani egy szomszédos, üres csövet, hogy ha a tárolója üres, valamint ezt az csövet később elhelyezni egy szomszédos, üres helyre. Pumpát olyan cső helyére tud letenni, aminek a bemenete és a kimenete is egy cső.

3.1.8 Szabotőr

Számítja azt az elemet, amin éppen áll. Szomszédos elemre léphet, hogyha ez lehetséges (aza, ha a szomszédos csövön esetleg nem áll másik játékos). A cisternák között direkt módon mozoghat.

Manipulálni tudja azt az elemet, amin éppen áll (csövet tud lyukasztani, illetve pumpát tud átállítani).

3.2 Statikus struktúra diagramok

fontos megjegyzések:

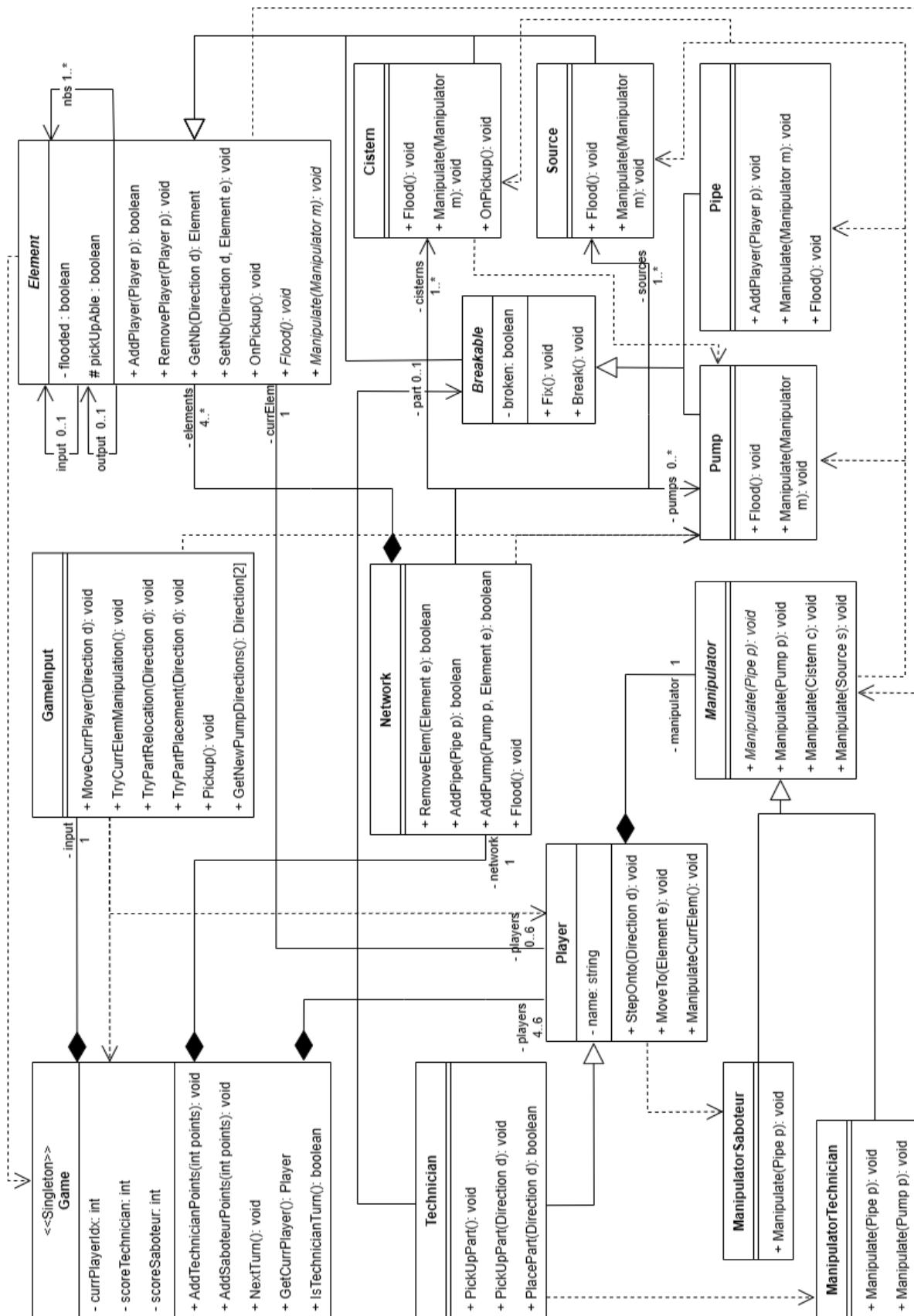
- A geometriát igyekeztünk eltávolítani. A Direction típus a modell szempontjából tetszőleges absztrakció lehet.
- A triviális (paraméter nélküli/egyparaméteres) setter/getter függvényeket nem jelöltük, viszont az "Osztályok leírása" szekcióban fel vannak ezek is sorolva.
- Ha egy ősnek van egy dependenciája, akkor az érvényes minden leszármazottra. A dependenciákat csak akkor jelöltük, ha nincsen erősebb kapcsolat az osztályok között (leszármazás, kompozíció, aggregáció, asszociáció).

osztálydiagram magyarázatok:

- GameInput-nak akár a Game része is lehetne, de számunkra ez egy fontos logikai elkülönítés:
 - GameInputba azok a függvények kerültek, amik az interfész szolgáltatják a View és Controller között (kommunikáció a felhasználóval).
 - Game-be közvetlenül pedig azok a függvények kerültek, amik a Controller részei.
- A visitor minta egy trade-off: a jelenlegi osztályhierarchiát bonyolítja, de új elemek potenciális megjelenésével a Player és leszármazott osztálya(i) könnyen túlságosan összetetté válhatnának, ha nem szerveznénk ki az elemek manipulációjának logikáját önálló osztályba.

3. Analízis modell kidolgozása

runtime_error



3.3 Osztályok leírása

3.3.1 Breakable

3.3.2 Cistern

- **Felelősség**

Amennyi csőből folyik víz ebbe az elembe, annyi pontot oszt a szerelők csapatának. Erről az elemről a játékosok közvetlen más ciszternákra tudnak lépni.

- **Ősosztályok**

Element

- **Metódusok**

- + void **OnPickUp()** : megpróbál pumpát adni a soron lévő játékos tárolójába.
- + void **Flood()** : Vízzel tölti fel magát, és pontot ad a szerelőknek. A pálya végpontjaként nem áramoltat más elemekbe vizet.
- + void **Manipulate(Manipulator m)** : A paraméterként kapott manipulátorral manipulálja ezt a konkrét elemet.

3.3.3 Element

- **Felelősség**

Player-eket tud felrakni, illetve eltávolítani magáról, és számon is tartja a rajta tartózkodókat. Tárolja a bemenet/kimenetét, valamint a szomszédjait is, ezeket tudja változtatni, illetve lekérdezni. Továbbá tárolja, hogy van-e víz benne.

- **Attribútumok**

- - **flooded: boolean** : Értéke igaz, ha víz van benne (jelenleg vizet tárol).
- # **pickUpAble: boolean** : Értéke igaz, ha az adott elem felvehető, különben hamis.
- - **players: Player[0..6]** : Azon játékosok halmaza, amelyek az adott elemen tartózkodnak
- - **nbs: Element[1..*]** : A vele szomszédos elemekre hivatkozik.
- - **input: Element[0..1]** : A bemenő szomszédra hivatkozik, ha van ilyen.
- - **output: Element[0..1]** : A kimenő szomszédra hivatkozik, ha van ilyen.

- **Metódusok**

- + void **AddPlayer(Player p)** : Felhelyez magára egy játékost, ha ez lehetséges a rajta állók száma szerint.
- + void **RemovePlayer(Player p)** : Eltávolítja a számoltartott játékosai közül azt, amelyiket paraméterként kapott.
- + Element **GetNb(Direction d)** : Megadott irányba lévő szomszédos elemét adja vissza, ha van ilyen.
- + void **SetNb(Direction d, Element e)** : Megadott irányú szomszédjának állítja be a paraméterként kapott elemet.
- + void **OnPickup()** : Ha egy elem támogatni akarja azt a funkcionálitást, hogy egy szerelő rajta állva egy part-ot kaphasson a tárolójába, akkor ezt a függvényt kell felülírnia. Alapértelmezett megvalósítása üres törzsű függvény (alapjáraton nem támogatják az elemek ezt a funkcionálitást).
- + void **Flood()** : Absztrakt metódus, amely a víz áramoltatásáért felelős.
- + void **Manipulate(Manipulator m)** : Absztrakt metódus. A paraméterként kapott manipulátorral manipulálja az elemet.
- + boolean **GetPickUpAble()** : Visszaadja, hogy az elem felvehető-e (pickUpAble és nincs benne víz)
- + int **GetNbCnt()** : A szomszédos elemek számát adja vissza.
- + int **GetPlayerCnt()** : A rajta tartózkodó játékosok számát adja vissza.
- + boolean **GetFlooded()** : Igazat ad vissza, ha az víz van benne (vagy víz folyik ki belőle), egyéb esetben hamisat.
- + Element **GetInput()** : Visszaadja a bemeneti elemét.
- + Element **GetOutput()** : Visszaadja a kimeneti elemét.
- + void **SetFlooded(boolean f)** : Az átadott érték szerint állítja be, hogy van-e víz jelenleg benne.
- + void **SetInput(Element e)** : A bemenetét az átadott elemre állítja be.
- + void **SetOutput(Element e)** : A kimenetét az átadott elemre állítja be.

3.3.4 Game

- **Felelősség**

A játékmenetet kezeli (vizsgálja, hogy véget ért-e a játék, valamint a köröket vezérli). Számoltartja és módosítja a csőrendszer felépítését (turn-önként véletlenszerűen pumpákat ront el, round-onként pedig ciszternákkal szomszédos üres helyekre új csöveket helyez). A turn-ök elején a forrásokból áramoltatja a vizet a csövekig, vagy ciszternáig. Pontokat oszt a csapatoknak, és számon is tartja azokat.

- **Attribútumok**

- - currPlayerIdx: int : A soron lévő játékos indexe a „players” gyűjteményben.
- - scoreTechnician: int : A szerelők pontszámát tárolja.
- - scoreSaboteur: int : A szabotőrök pontszámát tárolja.
- - input: Gameinput : A felhasználóval kommunikáló objektum.(Híd a View és a Model között.)
- - players: Player[4..6] : A játékban résztvevő játékosokat tárolja.
- - network: Network : A pálya elemeit tárolja, szortírozza.

- **Metódusok**

- + void **AddTechnicianPoints(int points)** : Ez a metódus ad a szerelőknek pontot.

- **+ void AddSaboteurPoints(int points)** : Ez a metódus ad a szabotőröknek pontot.
- **+ void NextTurn()** : A következő turn indítása (áramoltatja a vizet, pontokat oszt, véletlenszerűen pumpákat ront el, és ha véget ért egy teljes kör (round), akkor új csöveket teremt a ciszternák üres szomszédjai helyén). Ha az egyik csapat elérte a győzelemhez szükséges pontok számát, véget vet a játéknak.
- **+ Player GetCurrPlayer()** : Visszaadja a játékost, aki éppen soron van (akinek turnje van jelenleg).
- **+ boolean IsTechnicianTurn()** : Igazat ad vissza, ha éppen szerelő jön az adott körben.
- **+ GameInput GetInput()** : Visszaadja a felhasználóval kommunikáló objektumot.
- **+ Network GetNetwork** : Visszaadja a pálya elemeit számontartó objektumot.

3.3.5 GameInput

- **Felelősség**

A felhasználót és a játéket köti össze. Irányokat / célpontokat tud bekérni a felhasználótól. Játékosmozgatást, cső- és pumpalehelyezést, illetve csőfelvezést kezdeményezhet vele. Pumpát is tud adni a jelenlegi játékosnak, amennyiben egy szerelő, ciszernán áll és üres a tárolója.

- **Metódusok**

- + void **MoveCurrPlayer(Direction d)** : d irányba próbálja mozgatni az épp soron lévő játékost, arról az elemről, amelyiken éppen tartózkodik.
- + void **TryCurrElemManipulation()** : akkor hívandó függvény, amikor éppen azt az elemet szeretné manipulálni / interakcióba lépni vele (csövet lyukasztani / foltozni, pumpát javítani / átállítani vagy a következő ciszternára ugrani) a soron lévő játékos, amelyiken éppen áll.
- + void **TryPartRelocation(Direction d)** : akkor hívandú függvény, amikor a soron lévő játékos megpróbál egy tőle d irányba lévő szomszédos csövet felvenni, és a tárolójában eltárolni.
- + void **TryPartPlacement(Direction d)** : akkor hívandú függvény, amikor a soron lévő játékos megpróbálja a tárolt part-ját tőle d irányba lehelyezni.
- + void **Pickup()** : akkor hívandó függvény, amikor egy játékos egy part-ot próbálja a tárolójába tenni (nem a játéktérről, hanem közvetlenül odaadva). Ez csak akkor lesz sikeres, ha a játékos szerelő, üres a tárolója, és az elem amin áll, támogat ilyen funkcionálitást.
- + Direction[2] **GetNewPumpDirections()** : pumpa átállításához visszaadja a soron lévő játékos által bevitt irányokat.

3.3.6 Manipulator

- **Felelősség**

Absztrakt osztály, amely a Visitor tervezési mintát valósítja meg a leszármazottaival együtt. A leszármazottai egy játékos típusnak készítik el a „gazda” elem manipulálását megvalósító viselkedést minden elem esetére (pumpák, csövek, források, ciszternák).

A szerelők konkrét manipulátora például definiálja, hogy mit tud tenni egy szerelő, ha az előbb említett elemeken állva próbál interaktálni. A függvényei gyakran a kört is léptetik.

- **Metódusok**

- + void **Manipulate(Pipe p)** : Absztrakt metódus a játékosok cső manipulálásának leírására.
- + void **Manipulate(Pump p)** : Átállítja az átadott pumpát (GameInput-ot használja a bemenetért). Ezután véget ér a játékos köre (turn).
- + void **Manipulate(Cistern c)** : Átlépteti a jelenlegi játékost a következő ciszternára. Ezzel nem ér véget a játékos köre (turn).
- + void **Manipulate(Source s)** : Üres törzsű függvény, jelenleg a játékosok nem tesznek semmit a forráson állva. (A jövőbeli bővíthetőségre fenntartva, illetve a Dynamic Dispatch hibátlan működése miatt kell, hogy ne kelljen Type-checkingelni a hívóoldalon.)

3.3.7 ManipulatorSaboteur

- **Felelősség**

A szabotörök konkrét manipulátora, ami definiálja, hogy hogyan manipulálhatják az összes lehetséges „gazda” elemüket.

A csövek kilyukasztásának viselkedését valósítja meg. A többi viselkedés megfelel az űsbélivel.

- **Ősosztályok**

Manipulator

- **Metódusok**

- + void **Manipulate(Pipe p)** : Kilyukasztja az átadott p csövet. Utána pedig véget ér a jelenlegi játékos köre.

3.3.8 ManipulatorTechnician

- **Felelősség**

A szerelők konkrét manipulátora, ami definiálja, hogy hogyan manipulálhatják az összes lehetséges „gazda” elemüket.

A csövek javításának viselkedését valósítja meg, illetve a pumpák megjavításának viselkedésével helyettesíti a pumpaátállítást, ha az adott pumpa rossz.

A többi viselkedés megfelel az űsbélivel.

- **Ősosztályok**

Manipulator

- **Metódusok**

- + void **Manipulate(Pipe p)** : Megjavítja az átadott p csövet. Utána pedig véget ér a jelenlegi játékos köre.
- + void **Manipulate(Pump p)** : Megjavítja az átadott p pumpát, ha az elromlott, különben pedig átállítja (ez esetben meghívja az űsbéli megvalósítását a függvénynek).

3.3.9 Network

- **Felelősség**

Tárolja a pálya elemeit. Csövek és pumpák adhatók hozzá, a csöveket pedig el is lehet távolítani róla. Vizsgálja, hogy ezek a műveletek lehetségesek-e egyáltalán.

A benne tárolt elemek azok, amik ténylegesen meg is lesznek jelenítve a pályán.

Továbbá ez az osztály indítja el a vízfolyást forrásokból.

- **Attribútumok**

- - **elements: Element[4..*]** : A pálya összes elemét tárolja.
- - **cisterns: Cistern[1..*]** : Külön csoportosítja minden elem közül a ciszternákat ez a gyűjtemény.
- - **sources: Source[1..*]** : Külön csoportosítja minden elem közül a forrásokat ez a gyűjtemény.

- - **pumps: Pump[0..*]** : Külön csoportosítja minden elem közül a pumpákat ez a gyűjtemény.
- **Metódusok**
 - + **boolean RemoveElem(Element e)** : Megkísérli eltávolítani az átadott elemet a pályáról. Ha ennek semmi akadálya nem volt (pl. nem esne komponensekre a pálya, mint gráf), akkor igazzal tér vissza. Ellenkező esetben nem módosít semmi a pályán.
 - + **boolean AddPipe(Pipe p)** : Az átadott csövet megkísérli hozzáadni a pályához. Ennek sikerességét adja vissza.
 - + **boolean AddPump(Pump p, Element e)** : Az átadott pumpával kísérli meg felülírni a másik átadott elemet a pályán. Ennek sikerességét adja vissza.
 - + **void Flood()** : Felapasztja az összes vizet a pályáról, majd minden forrásból elindítja a vizet, aminek következtében el lesz árasztva vízzel a pálya, és pontokat fognak kapni a csapatok.
 - + **Element[4..*] GetElements()** : Visszaadja a pálya minden elemét.
 - + **Pump[0..*] GetPumps()** : Visszaadja a pályán lévő pumpákat.
 - + **Cistern[1..*] GetCisterns()** : Visszaadja a pályán lévő ciszternákat.

3.3.10 Pipe

- **Felelősség**

Olyan elromolható, felvehető elem, amelyen legfeljebb egy játékos tartózkodhat.

Ki lehet lyukasztani, a lyukas csövet pedig meg is lehet javítani. Ha a cső nem lyukas akkor a kimenetéhez vizet tud juttatni.

Ha lyukas, vagy a kimenete üres, akkor a szabotőrök pontot kapnak, amikor víz érkezik belé.

- **Ősosztályok**

Element → Breakable

- **Metódusok**

- + void **AddPlayer(Player p)** : Felhelyezi az átadott játékost magára, ha nem áll rajta már más valaki.
- + void **Manipulate(Manipulator m)** : Az átvett manipulátorral manipulálja ezt a konkrét elemet.
- + void **Flood()** : Vízzel tölti meg magát. Ha lyukas a cső vagy nincs output-ja akkor pontot kapnak a szabotőrök, különben pedig hívja tovább az output-ján a Flood() függvényt (ereszti tovább a vizet).

3.3.11 Player (Saboteur)

- **Felelősség**

A Player osztály a szabotőr osztállyal ekvivalens, tehát a szabotőrök csapatának játékosai Player típusúak. Eltárolja a játékos nevét, az elemet, amelyen éppen áll, és a manipulátorát. Tud mozogni irányokba, és manipulálni tudja a „gazda” elemét.

- **Attribútumok**

- - currElem: Element : Az adott elem, amelyen a játékos tartózkodik
- - name: string : A játékos neve, azonosítója
- - manipulator: Manipulator : A játékos által használt manipulator objektum a „gazda” elemmel való interakciók lekezeléséhez.

- **Metódusok**

- + void **MoveTo(Element e)** : Az átadott elemre helyezi a játékost, ha ez lehetséges.
- + void **StepOnto(Direction d)** : A játékost a megadott irányban lévő elemre lépteti, ha ez lehetséges.
- + void **ManipulateCurrElem()** : Manipulálja azt az elemet, amelyen éppen tartózkodik. A manipulátora határozza meg, hogy mely „gazda” elem típus esetén mit tesz.
- + Element **GetCurrElem()** : Visszaadja az elemet, amelyen a játékos éppen tartózkodik.

3.3.12 Pump

- **Felelősség**

Olyan elromolható, felvehető elem, amely vizet képes átereszteni a bemenetként beállított csövéről a kimenetként beállított csövébe. Pontot oszt a szabotöröknek, ha nincsen kimenete, amikor vizet ereszte át.

- **Ósosztályok**

Element → Breakable

- **Metódusok**

- + **void Flood()** : Ha a pumpa el van romolva, nem tesz semmit. Ha nincs elromolva akkor vízzel tölti fel magát. Ha nincsen outputja, akkor a szabotörök pontot kapnak, de ha van outputja, akkor az outputra hívja tovább a Flood() függvényt.
- + **void Manipulate(Manipulate m)** : Az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.13 Source

- **Felelősség**

Olyan elem, amely vizet juttat a vele szomszédos elemekbe, és soha sincsen bemenete.

- **Ósosztályok**

Element

- **Metódusok**

- + **void Flood()** : Vízzel tölti fel magát, és minden nem lyukas szomszédjára hívja tovább a Flood() függvényt, azaz ereszti tovább a vizet. Ha nem tudja ezt megtenni az adott szomszédja felé, mert lyukas, vagy nincs arra szomszédja, akkor pontot ad a szabotöröknek.
- + **void Manipulate(Manipulator m)** : Az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.14 Technician

- **Felelősség**

Olyan játékos, aki csövek lyukasztása helyett javítani tudja őket, illetve pumpákat is. Emellett képes vagy egy csövet, vagy egy pumpát tárolni magánál, ami el is tud helyezni a megfelelő feltételek fennállása esetén.

- **Ósosztályok**

Player

- **Attribútumok**

- - **part: Breakable[0..1]** : Az az elem, amelyet a szerelő felvett a pályáról.

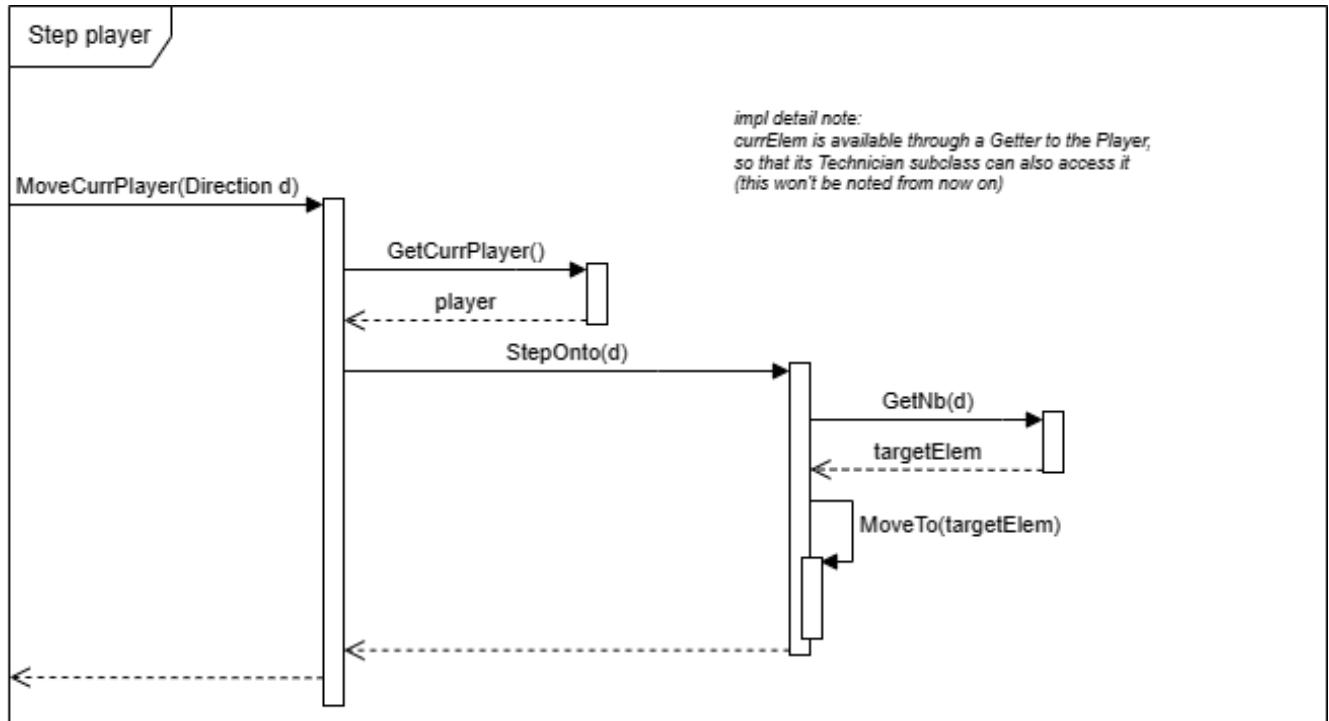
- **Metódusok**

- + **void PickUpPart()** : Megkísérel felvenni egy part-ot a jelenlegi elemtől.
- + **void PickUpPart(Direction d)** : Megkísérli felvenni a d irányban lévő part-ot.

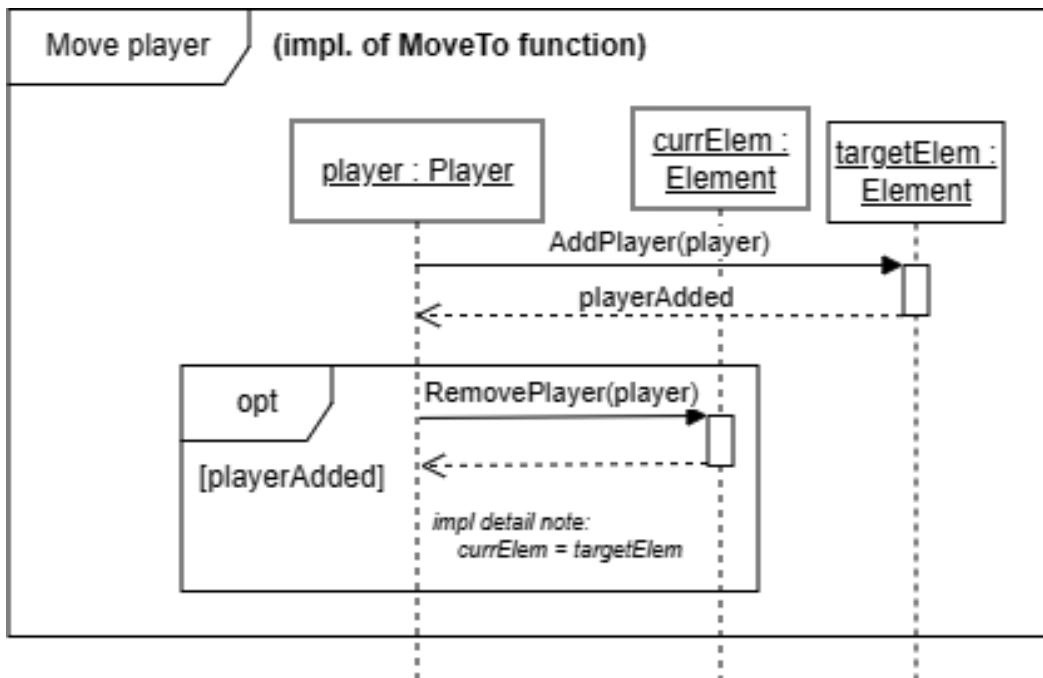
- + **boolean PlacePart(Direction d)** : Megkísérli elhelyezni a tárolt part-ját d irányba. A művelet sikerességevel tér vissza.
- + **Breakable GetPart()** : Visszaadja azt a Breakable-t („part” attribútumot), ami a szerelőnél van.
- + **void SetPart(Breakable b)** : Az átadott Breakable-re állítja a „part” attribútumot.

3.4 Szekvencia diagramok

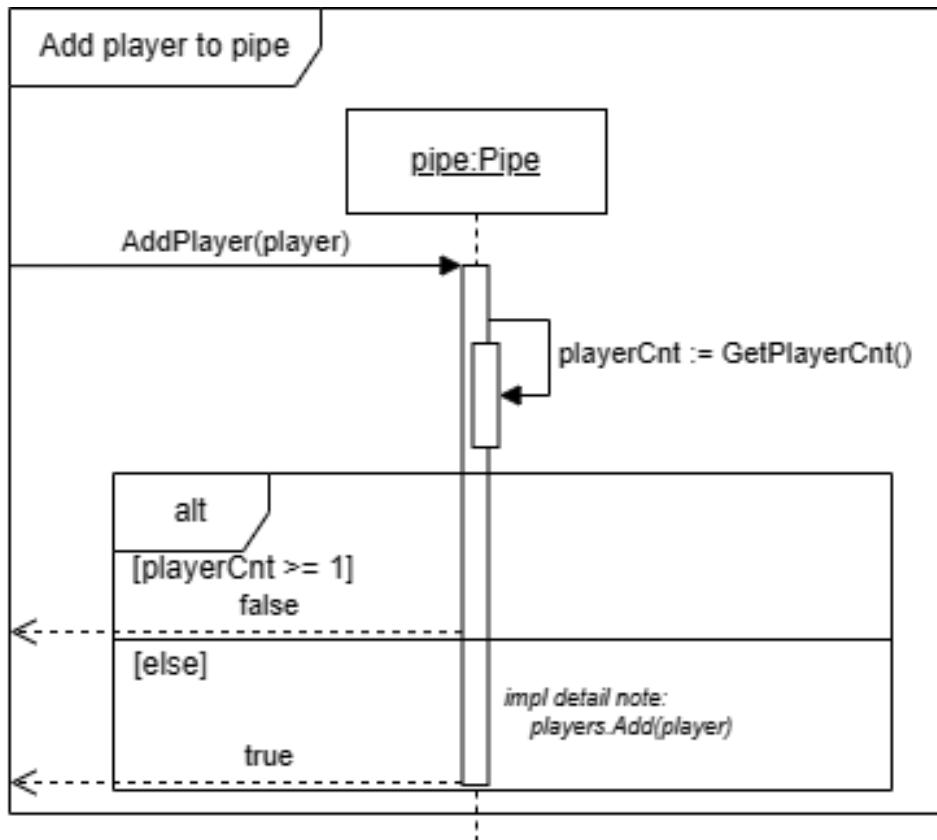
3.4.1 Step Player



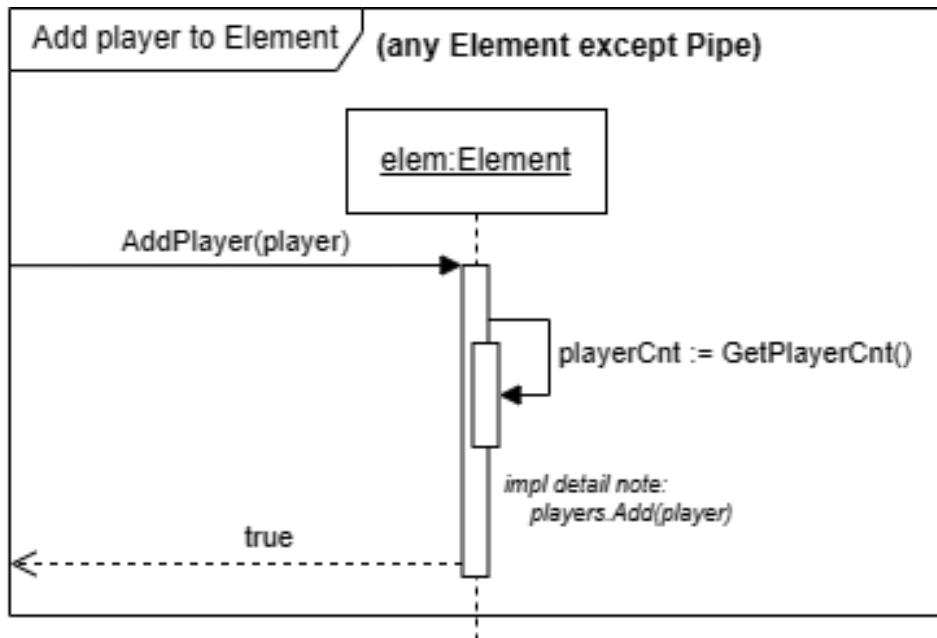
3.4.2 Move Player



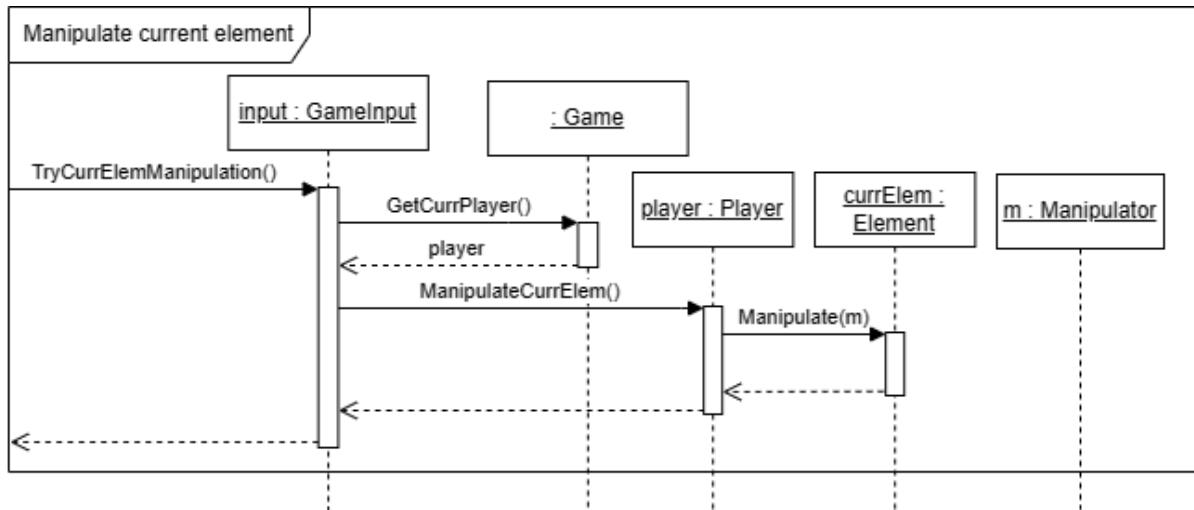
3.4.3 Add Player to Pipe



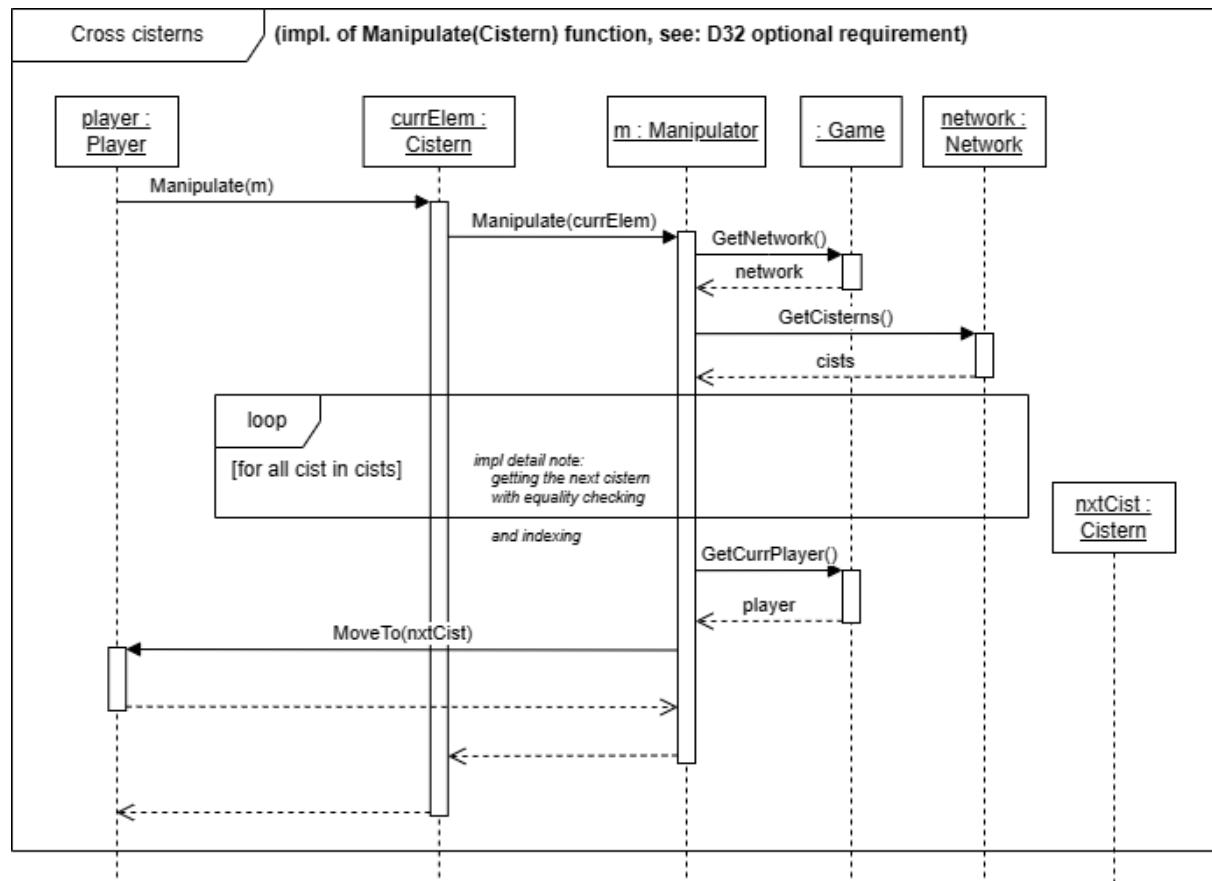
3.4.4 Add Player to Element



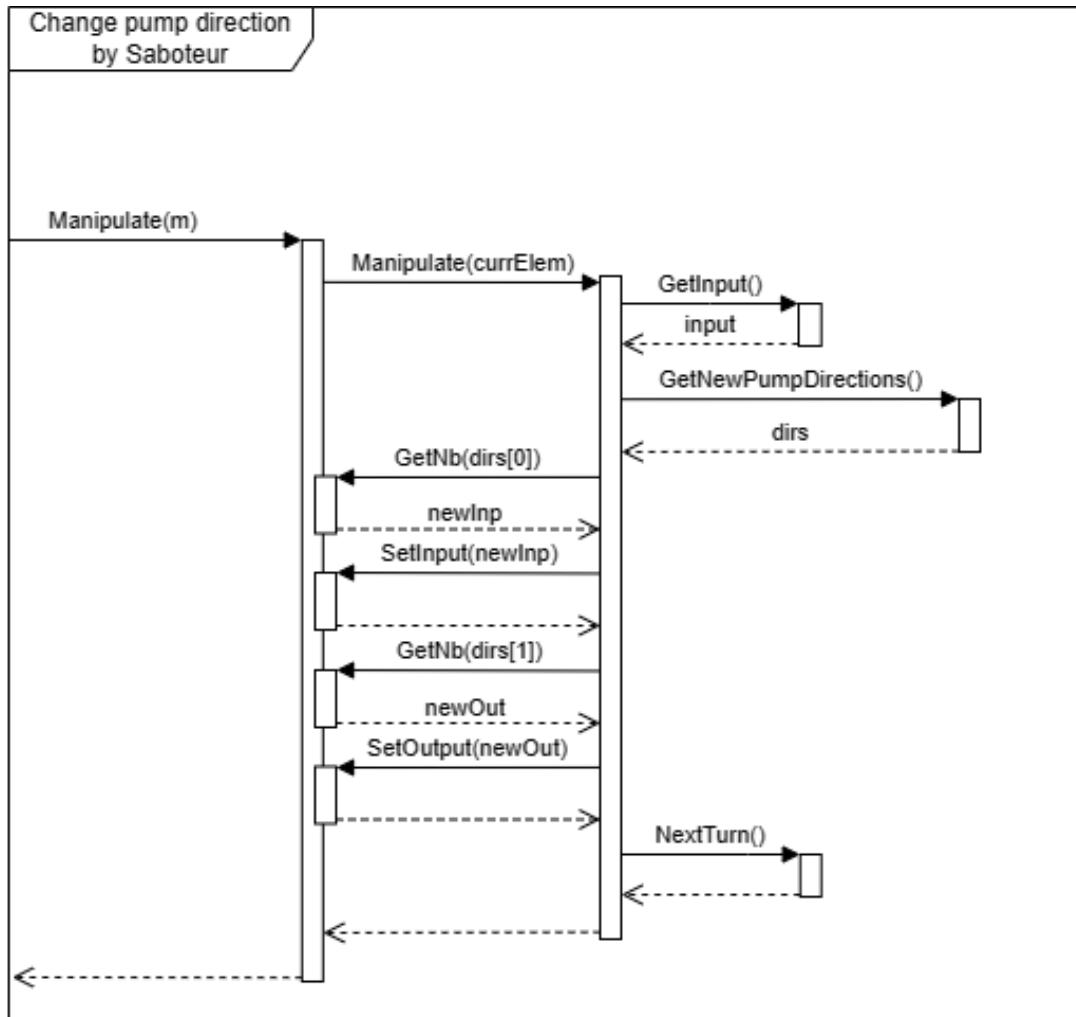
3.4.5 Manipulate Current Element



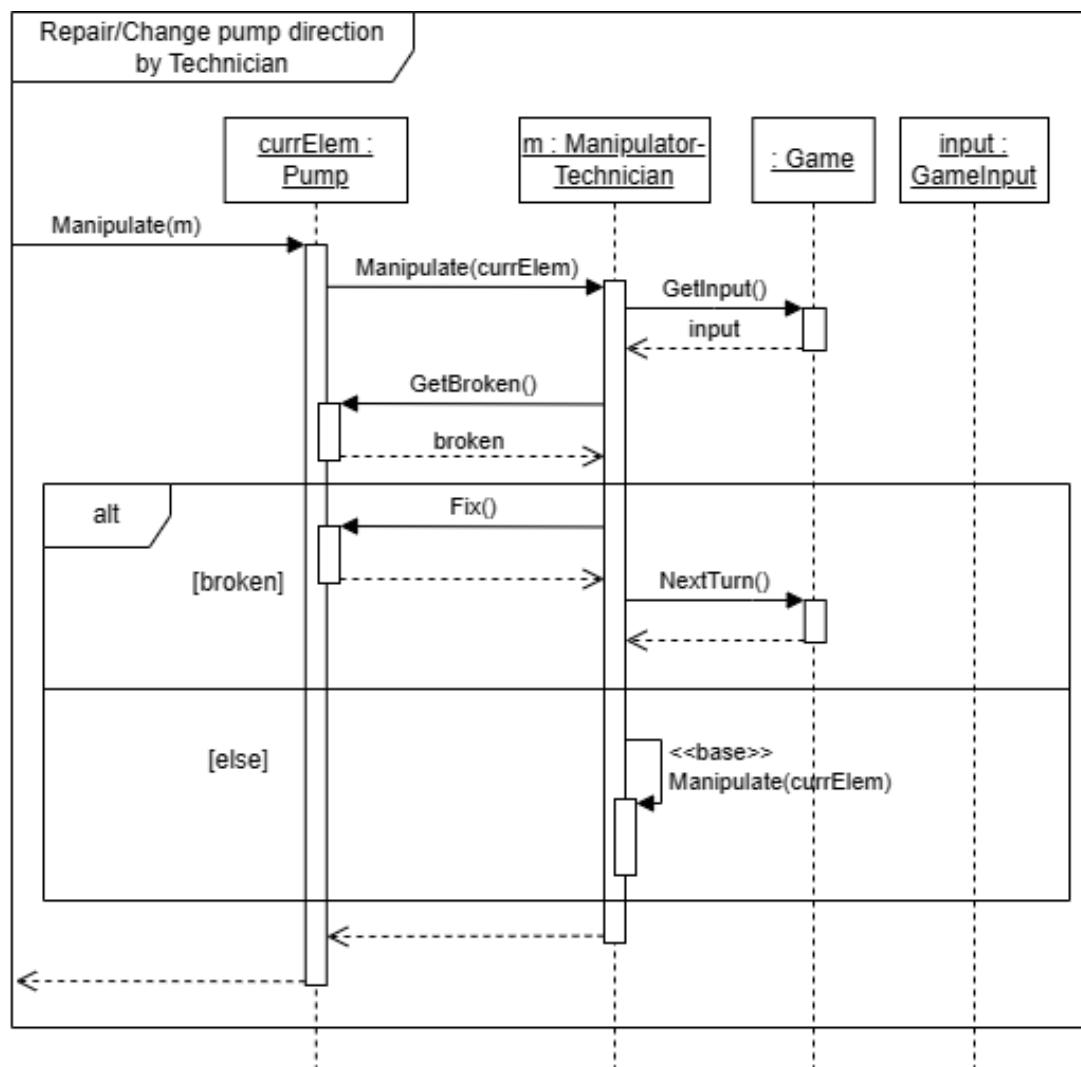
3.4.6 Cross Cisterns



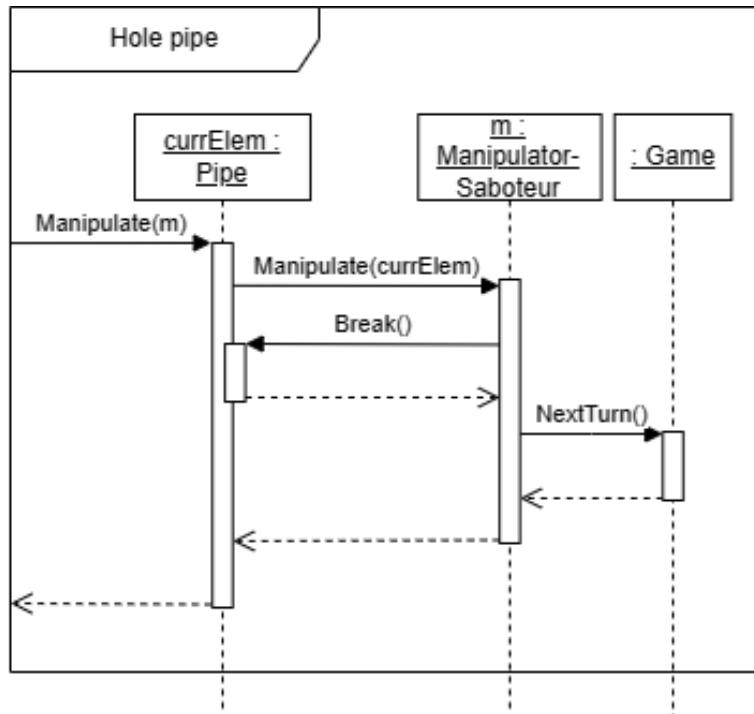
3.4.7 Change Pump Direction (by Saboteur)



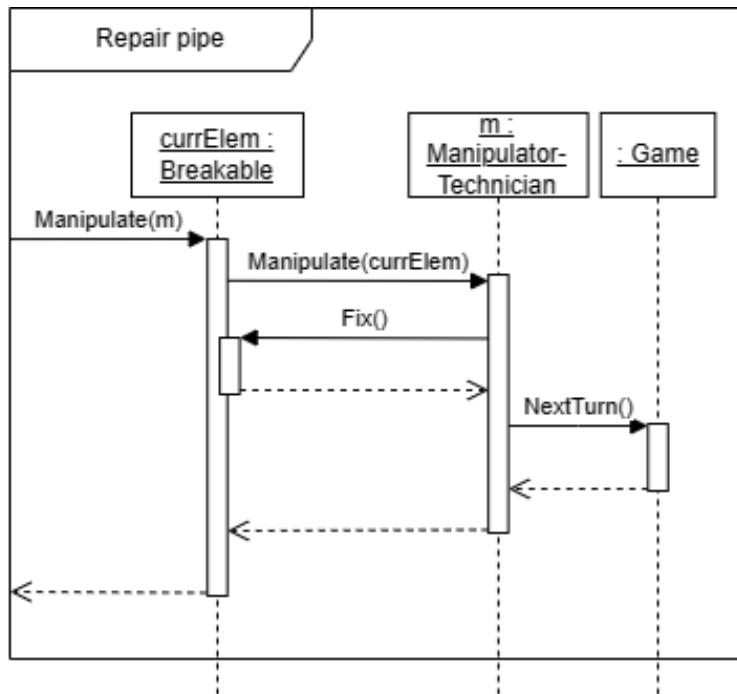
3.4.8 Repair or Change Pump Direction (by Technician)



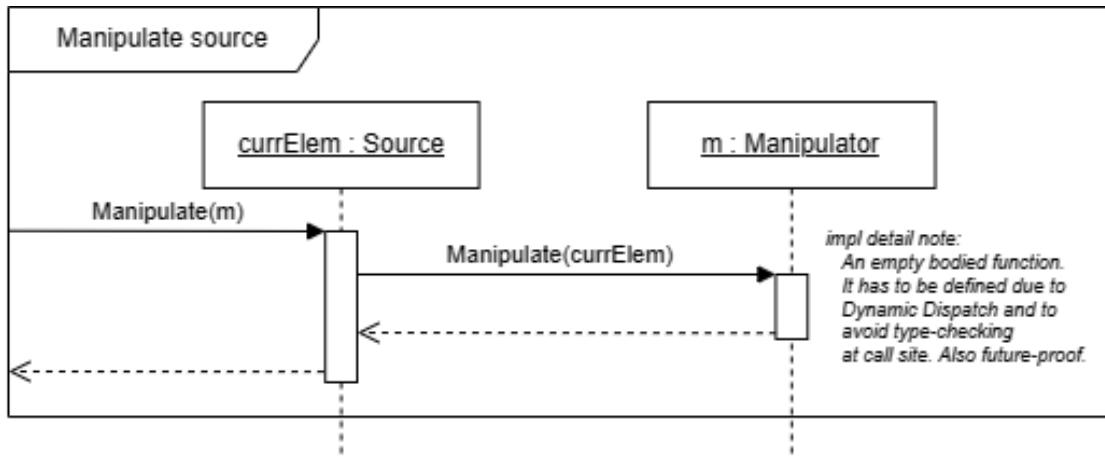
3.4.9 Hole Pipe



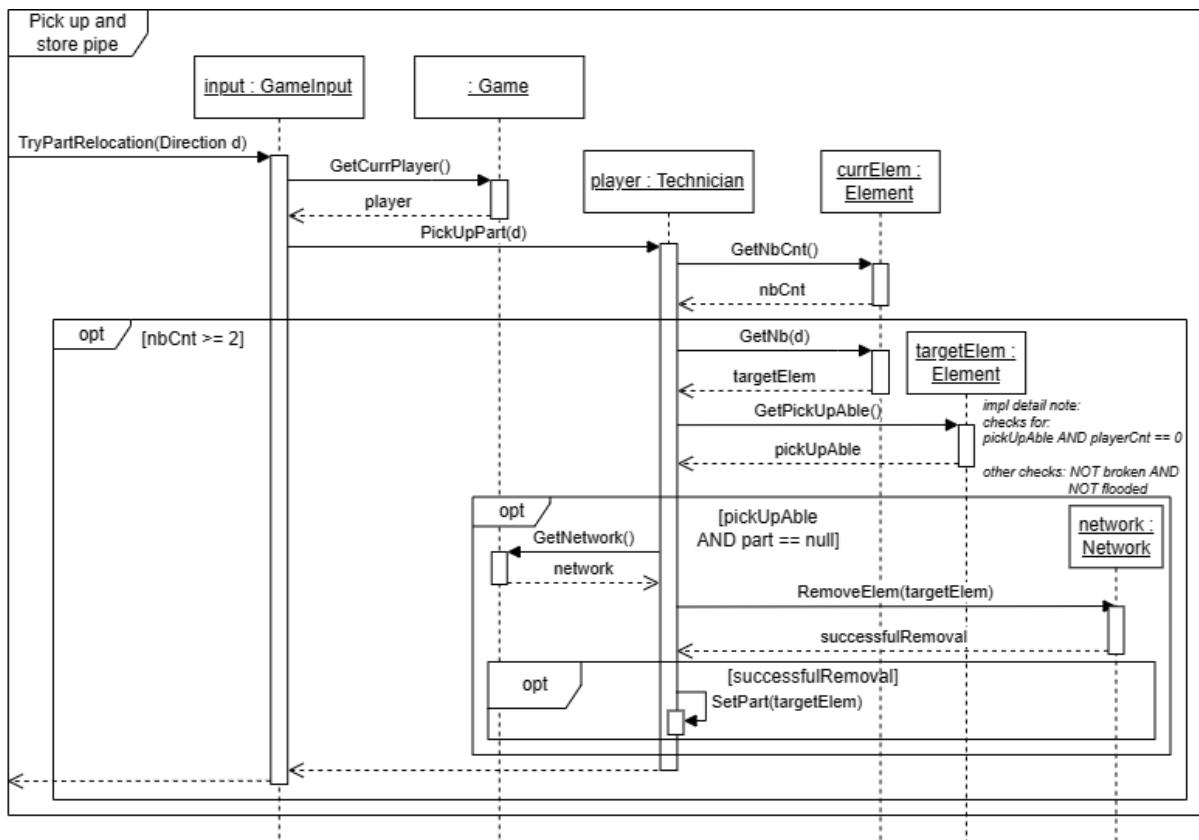
3.4.10 Repair Pipe



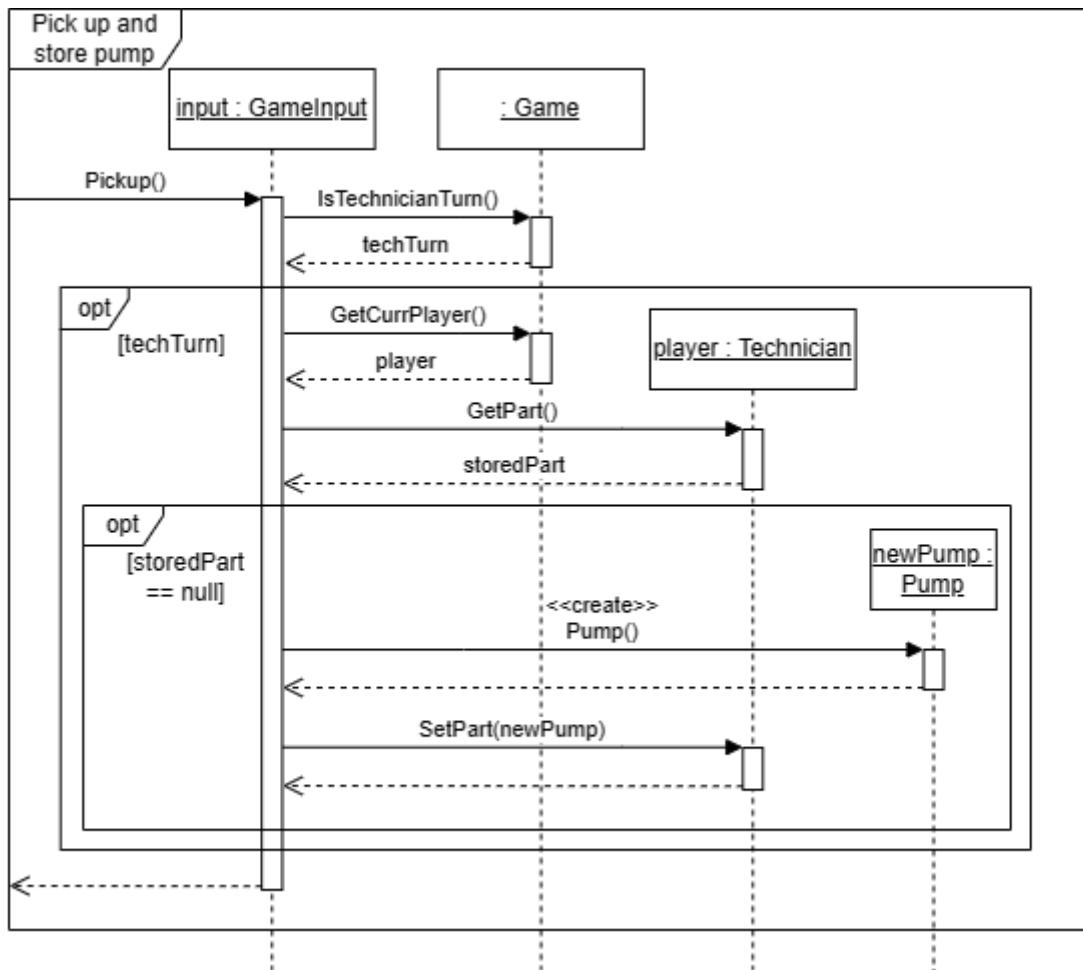
3.4.11 Manipulate Source



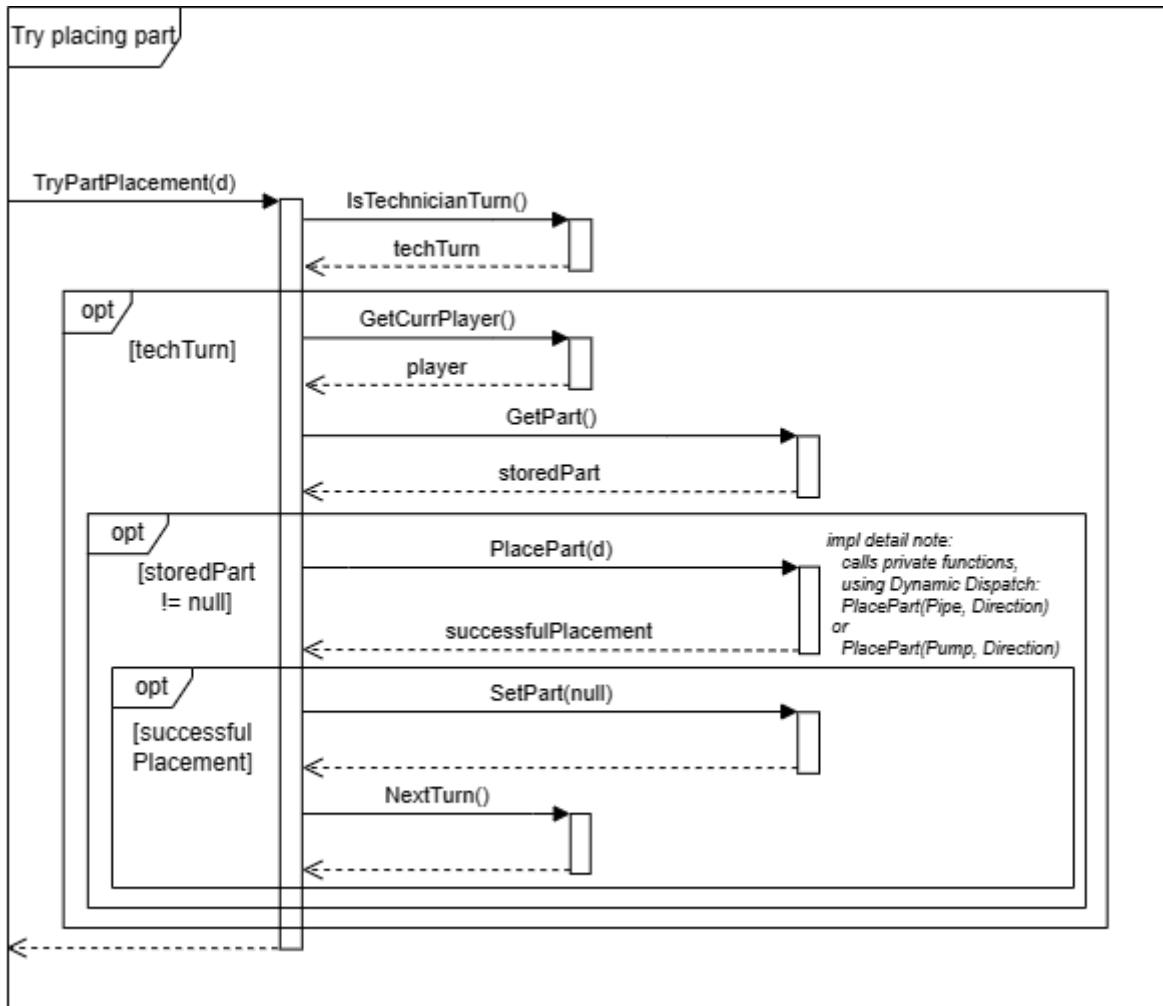
3.4.12 Pick Up and Store Pipe



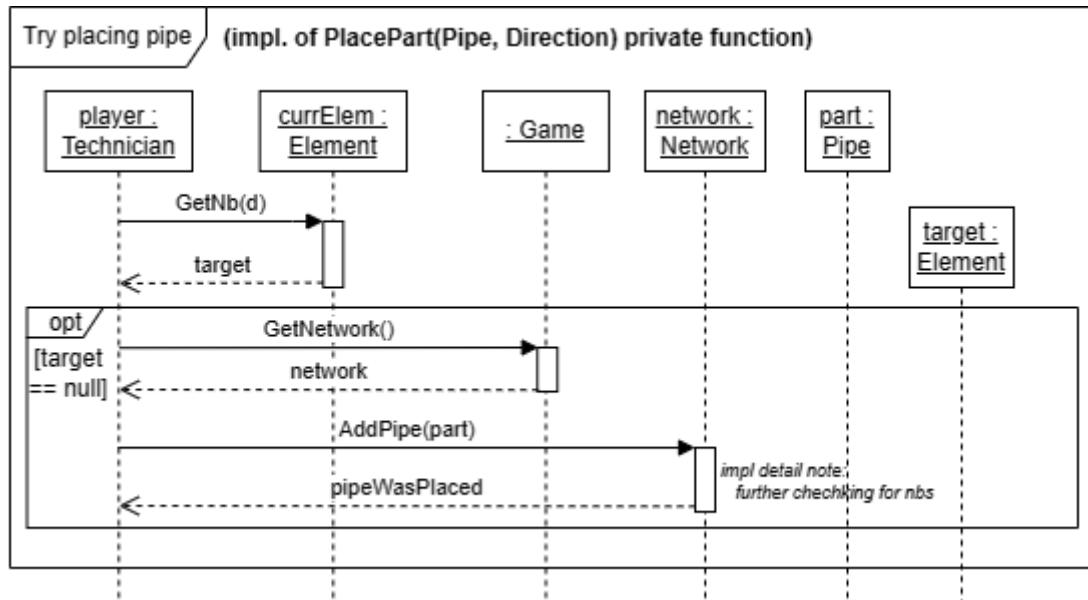
3.4.13 Pick Up and Store Pump



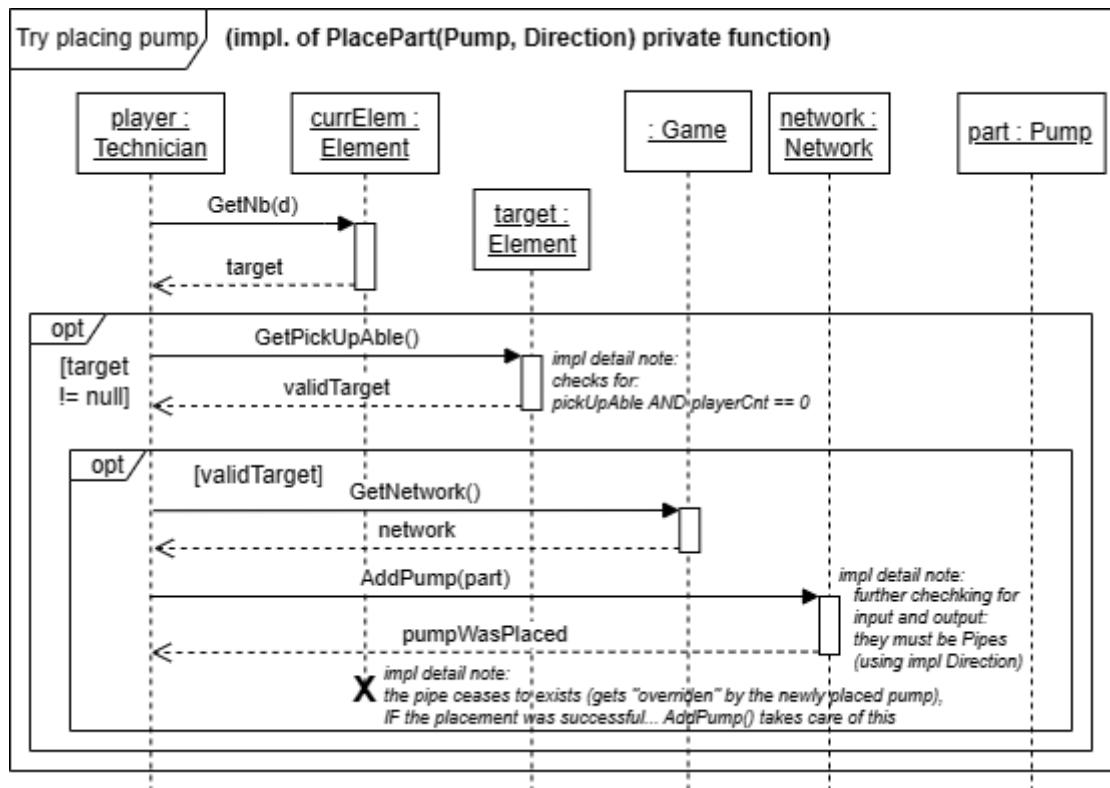
3.4.14 Try Placing Part



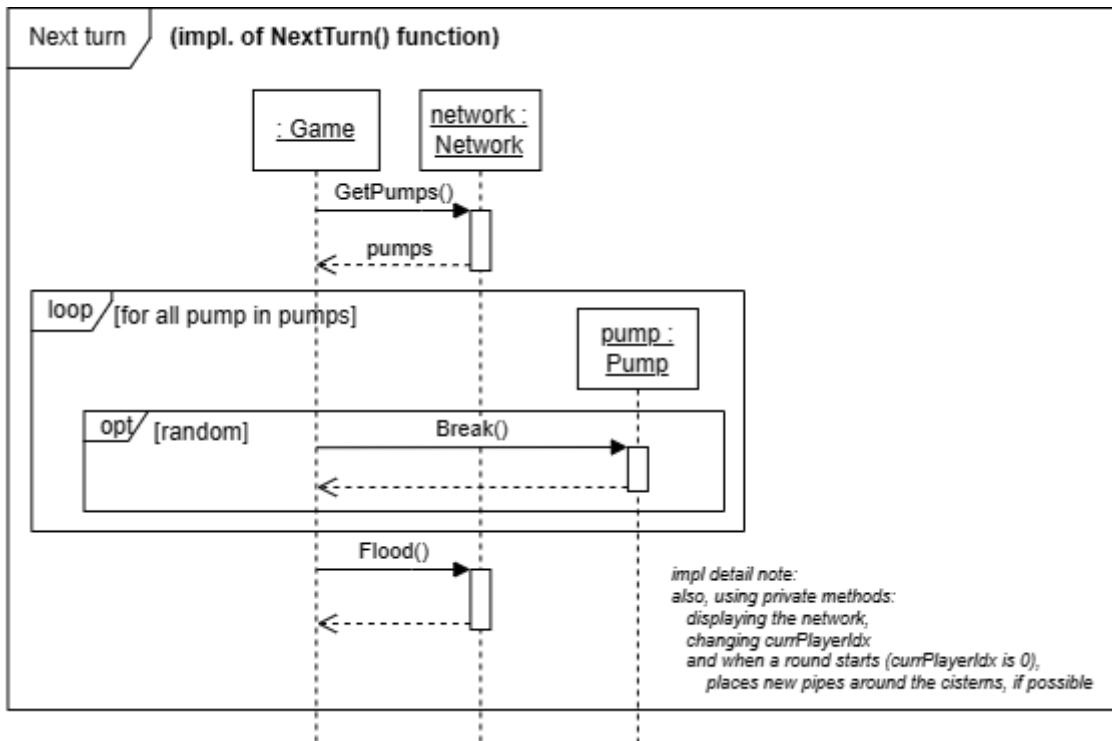
3.4.15 Try Placing Pipe



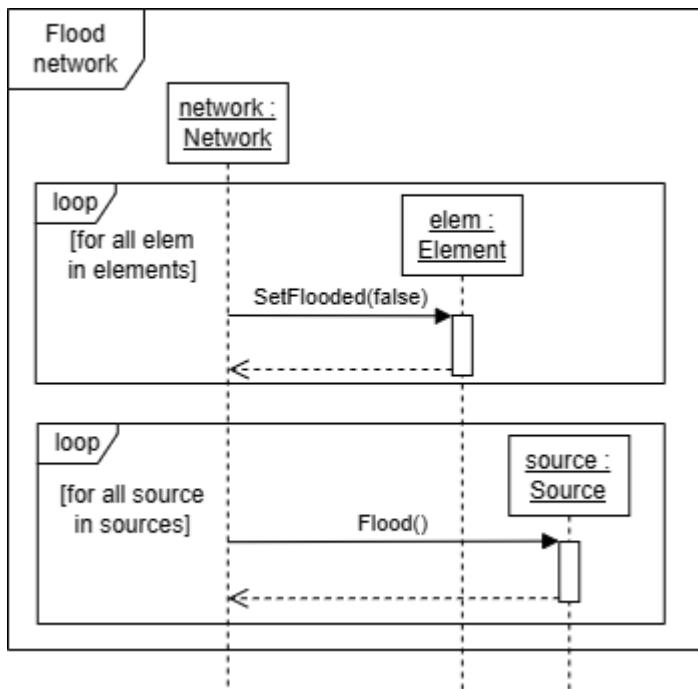
3.4.16 Try Placing Pump



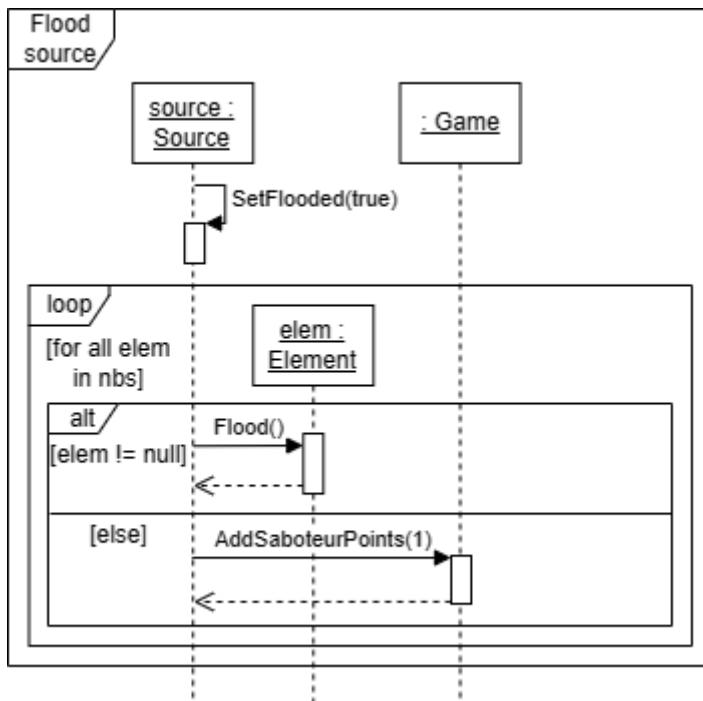
3.4.17 Next Turn



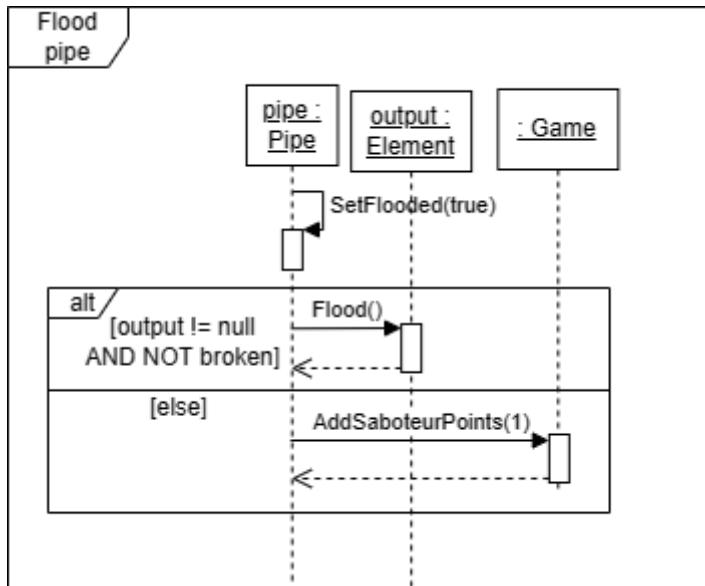
3.4.18 Flood Network



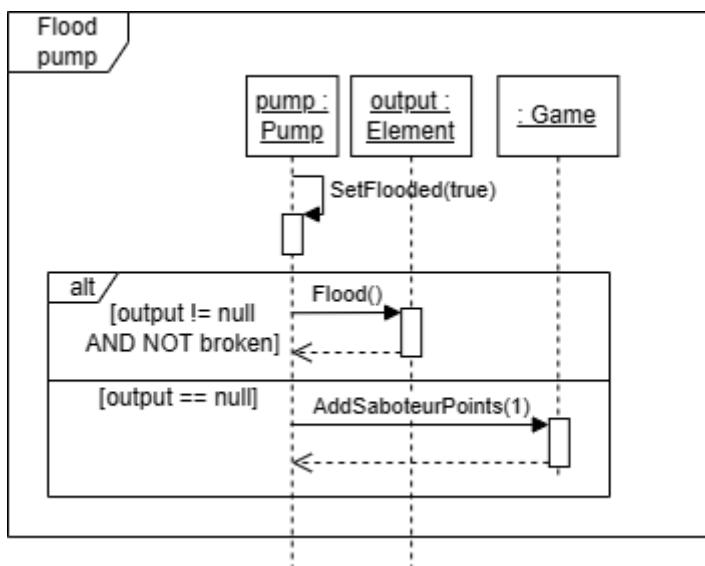
3.4.19 Flood Source



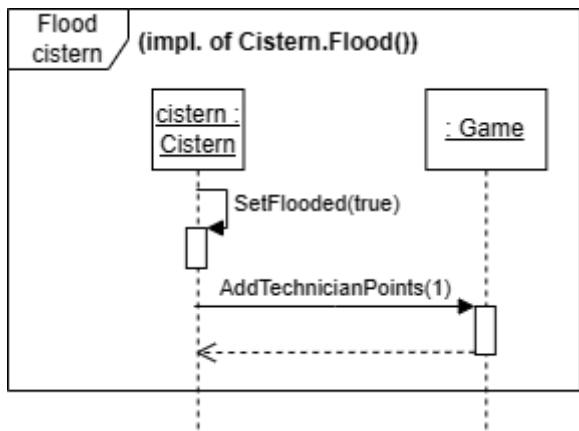
3.4.20 Flood Pipe



3.4.21 Flood Pump

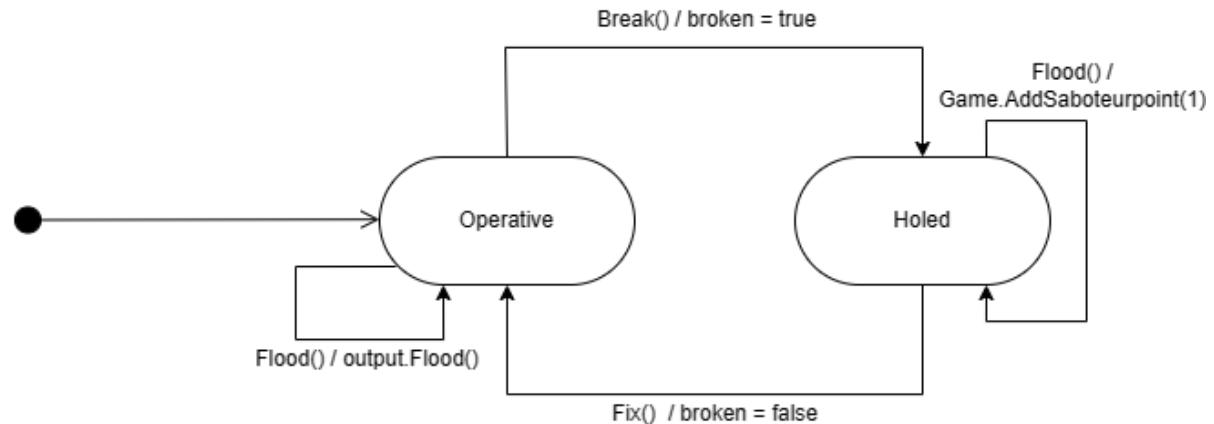


3.4.22 Flood Cistern

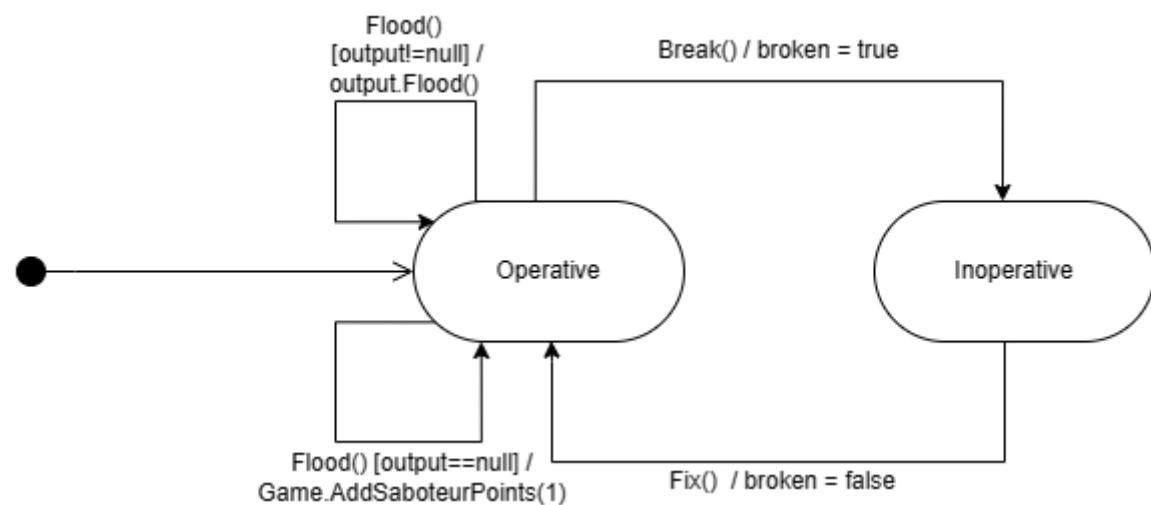


3.5 State-chartok

3.5.1 Pipe States



3.5.2 Pump States



3.6 Napló

Kezdet	Időtartam	Részttvevők	Leírás
2023.03.22. 19:00	0,5 óra	Mizser Váradi Tepliczky Fekete Sasvári	<p>Értekezlet. Döntés:</p> <ul style="list-style-type: none"> • Mizser és Sasvári javítja az osztálydiagramot és a szekvenciadiagramokat • Fekete, Tepliczky és Sasvári elkészíti a 2 hiányzó szekvenciadiagramot • Váradi javítja a dokumentum formai és tartalmi hiányosságait <hr/> <p>Diagram javítások Határidő: 2023.03.25. 22:00. Elkészült: 2023.03.25. 20:00</p> <p>Kész dokumentum (formázással) Határidő: 2023.03.27. 0:00 Elkészült: 2023.03.26 21:00</p>
2023.03.23. 18:00	3 óra	Mizser Sasvári	<p>Osztálydiagram és szekvenciadiagramok hibáinak feltárása:</p> <ul style="list-style-type: none"> • cső életvonalának megszakadása amikor felülíródik a lerakott pumpával • "Repair/Change pump direction by Technician"-ben a felesleges GetPumpDirections hívás törlése • GetCurrElem getter hozzáadása Playerhez, ennek jelölése szekvencián megjegyzéssel • MoveTo(targetElem) felvétele osztálydiagramra • GetPickUpAble függvény leírásának megváltoztatása
2023.03.23 20:00	2 óra	Mizser	<p>Az előzőek tényleges módosítása a diagramokon. További javítások:</p> <ul style="list-style-type: none"> • „osztálydiagram magyarázatok” rész megírása • nem egyértelmű szekvencia diagramokra az illusztrált függvény nevének kiírása • konzisztens függvénynevek
2023.03.25. 18:30	1,5 óra	Tepliczky, Fekete, Sasvári	Pipe States és Pump States állapotdiagramok elkészítése. Korábbi javítások áttekintése.
2023.03.26. 14:00	1 óra	Váradi	<p>A dokumentum formai és tartalmi javítása:</p> <ul style="list-style-type: none"> • A hiányzó oldalszámok pótlása • Objektumkatalógus pontosítása

2023.03.26. 17:00	1 óra	Váradi	A dokumentum tartalmi javítása: • Pontosítás és korrekció az Objektumkatalógus szövegében és az osztályleírásokban.
2023.03.26. 19:00	2 óra	Váradi	A dokumentum tartalmi és formai javítása: • További apróbb korrekciók az osztályok leírásában. • Szekvenciadiagramok frissítése. • Állapotdiagramok frissítése. • Dokumentum tördelése és formázása.

3. Analízis modell (II. változat)

5 – runtime_error

Konzulens:
Dobos-Kovács Mihály

Csapattagok

Mizser Ádám Zoltán	SHKGZW	mizser.adam@gmail.com
Tepliczky Olivér Zoltán	WB6LC5	tepliczkyo@edu.bme.hu
Fekete Álmos Valér	KR5WPC	feketealmos0@gmail.com
Váradi Kristóf	BP17IB	kristofvaradi@edu.bme.hu
Sasvári Szabolcs Attila	TWOZG6	szabolcs.attila.sasvari@edu.bme.hu

2023.03.26.

Analízis modell kidolgozása

3.1 Objektum katalógus

fontos megjegyzés: A Game (játék) objektum a játékvezérlő kontrollert valósítja meg. A nem játékosokhoz köthető használati esetekért felel, illetve ez a híd a View és Model között.

Bár nem képezi az analízis modell részét, meg kell jelenítenünk, mert számos szekvenciadiagramhoz elengedhetetlen, valamint ez a tervezést is jelentősen megkönnyíti.

3.1.1 Játék

A játékmenetet kezeli (vizsgálja, hogy véget ért-e a játék, valamint a köröket vezérli). Számítja és módosítja a csőrendszer felépítését (turn-önként véletlenszerűen pumpákat ront el, round-onként pedig ciszternákkal szomszédos üres helyekre új csöveket helyez). A turn-ök elején a forrásokból áramoltatja a vizet a csövekig, vagy ciszternáig. Pontokat oszt a csapatoknak, és számon is tartja azokat.

Irányokat / célpontokat tud bekérni a felhasználótól. Játékosmozgatást, cső- és pumpalehelyezést, illetve csőfelvételt kezdeményezhet vele. Pumpát adhat a ciszternán található szerelőnek, ha üres a tárolója.

3.1.2 Csőrendszer

Számítja a játék elemeit (csöveket, pumpákat, forrásokat és ciszternákat). Ezek közül eltávolíthat csöveket, ha a csőrendszer ettől még összefüggő marad. Az eltávolított csövet odaadja annak a szerelőnek, aki az eltávolítást kezdeményezte. Kezeli a csövek és pumpák elhelyezését, valamint az erre irányuló utasítások helyességét.

3.1.3 Forrás

Olyan elem, amely vizet juttat a vele szomszédos csövekbe, azaz a számított kimeneteibe, illetve a sivatagba, ha nincs szomszédja az adott irányban. Ahányszor előfordul az utóbbi, annyi pontot oszt a szabotőrök csapatának.

Játékosokat tárol és távolít el az adott pozíóról.

3.1.4 Ciszterna

Olyan elem, amely számítja a bemeneteit. Amennyi csőből folyik víz az adott ciszternába, annyi pontot oszt a szerelők csapatának.

Számítja a játékosokat, akik rajta állnak. Játékosokat tud felenni, illetve eltávolítani magáról.

3.1.5 Pumpa

Olyan elem, amely számítja a bemenetét, a kimenetét és a szomszédait. A szomszédai közül be tudja állítani, hogy melyik a bemenet és melyik a kimenet. Vizet képes juttatni a bemeneti csövéből, önmagán keresztül (ő maga is tárol vizet) a kimeneti csövébe. El tud romlani, és ekkor nem képes víz átengedésére. Amikor víz megy át rajta, pontot oszthat a szabotőrök csapatának, ha a kimenete egy szabad végű / lyukas cső vagy sivatag (nincs kimenete).

Számítja a játékosokat, akik rajta állnak ezzel együtt játékosokat tud felenni, illetve eltávolítani magáról.

3.1.6 Cső

Olyan elem, amely számítartja a bemenetét és kimenetét, és az előbbiből az utóbbiba vizet képes juttatni. Tárolja a benne lévő vizet. Ki lehet lyukasztani, és meg is lehet javítani.

Lyukas állapotában nem képes vizet átengedni magán. A víz átfolyásakor pontot oszt a szabotöröknek, amennyiben a cső lyukas, vagy kimenete a sivatag.

Számítartja a játékost, aki rajta áll, valamint egy játékost tud felenni, illetve eltávolítani magáról (egyszerre egy játékos lehet egy csövön).

3.1.7 Szerező

Számítartja azt az elemet, amin éppen áll. Szomszédos elemre léphet, hogyha ez lehetséges (aza, ha a szomszédos csövön esetleg nem áll másik játékos). A cisternák között direkt módon mozoghat.

Manipulálni tudja az elemet, amin éppen áll (csövet és pumpát tud javítani, illetve működő pumpát tud átállítani).

A tárolója tárolhat vagy egy csövet, vagy egy pumpát. El tud távolítani egy szomszédos, üres csövet, hogy ha a tárolója üres, valamint ezt az csövet később elhelyezni egy szomszédos, üres helyre. Pumpát olyan cső helyére tud letenni, aminek a bemenete és a kimenete is egy cső.

3.1.8 Szabotőr

Számítartja azt az elemet, amin éppen áll. Szomszédos elemre léphet, hogyha ez lehetséges (aza, ha a szomszédos csövön esetleg nem áll másik játékos). A cisternák között direkt módon mozoghat.

Manipulálni tudja azt az elemet, amin éppen áll (csövet tud lyukasztani, illetve pumpát tud átállítani).

3.2 Statikus struktúra diagramok

fontos megjegyzések:

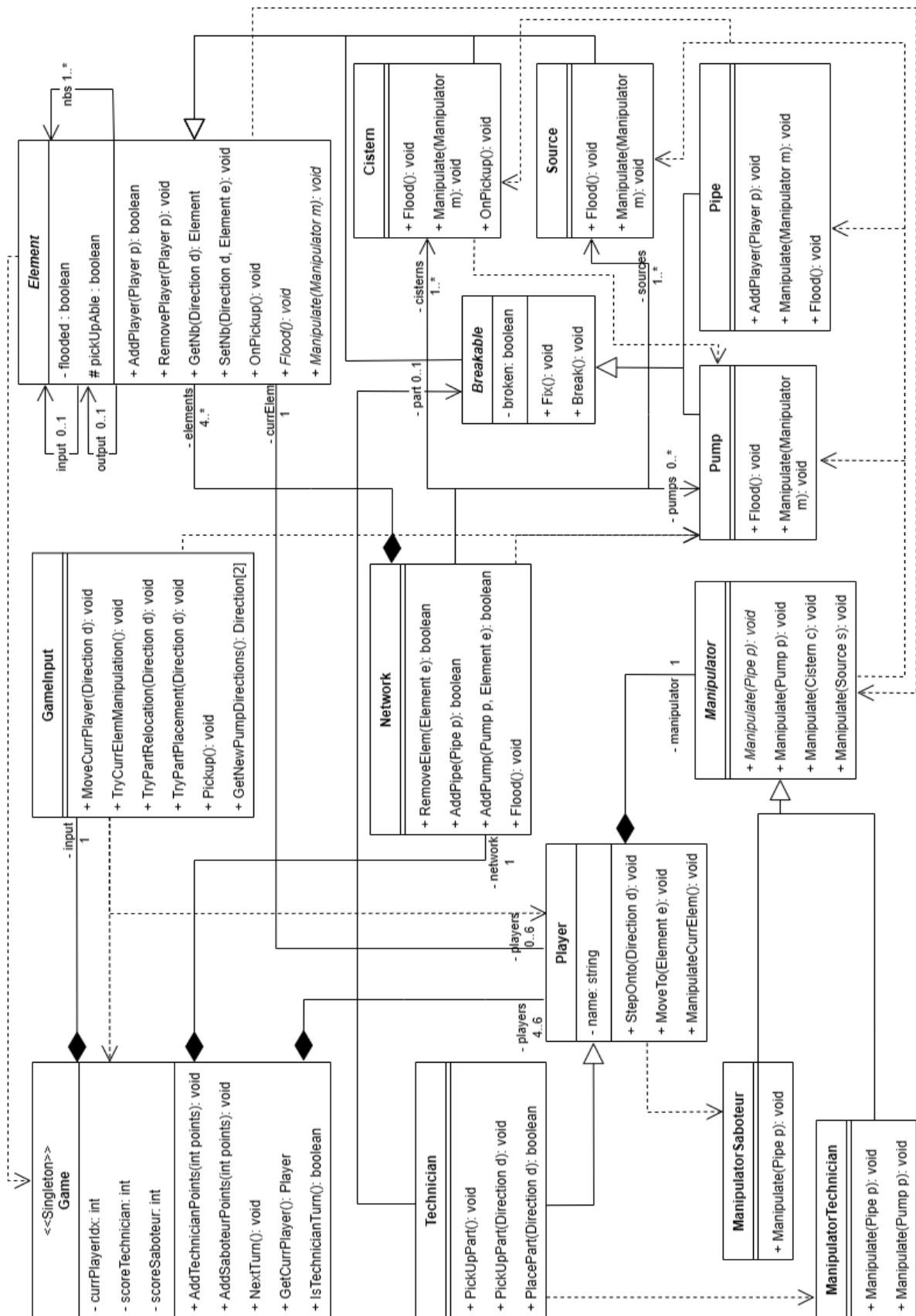
- A geometriát igyekeztünk eltávolítani. A Direction típus a modell szempontjából tetszőleges absztrakció lehet.
- A triviális (paraméter nélküli/egyparaméteres) setter/getter függvényeket nem jelöltük, viszont az "Osztályok leírása" szekcióban fel vannak ezek is sorolva.
- Ha egy ősnek van egy dependenciája, akkor az érvényes minden leszármazottra. A dependenciákat csak akkor jelöltük, ha nincsen erősebb kapcsolat az osztályok között (leszármazás, kompozíció, aggregáció, asszociáció).

osztálydiagram magyarázatok:

- GameInput-nak akár a Game része is lehetne, de számunkra ez egy fontos logikai elkülönítés:
 - GameInputba azok a függvények kerültek, amik az interfész szolgáltatják a View és Controller között (kommunikáció a felhasználóval).
 - Game-be közvetlenül pedig azok a függvények kerültek, amik a Controller részei.
- A visitor minta egy trade-off: a jelenlegi osztályhierarchiát bonyolítja, de új elemek potenciális megjelenésével a Player és leszármazott osztálya(i) könnyen túlságosan összetetté válhatnának, ha nem szerveznénk ki az elemek manipulációjának logikáját önálló osztályba.

3. Analízis modell kidolgozása

runtime_error



3.3 Osztályok leírása

3.3.1 Breakable

- **Felelősség**

Ez az elem elromolhat, és értelemszerűen meg is lehet javítani. Ilyen elemet tudnak tárolni a szerelők (más néven part).

- **Ősosztályok**

Element

- **Attribútumok**

- - **broken: boolean** azt jelzi, hogy az adott elem hibás-e vagy sem, azaz át tud-e ereszteni magán vizet, vagy sem.

- **Metódusok**

- + **void Fix()** : Hibás elem megjavítására szolgáló függvény.
- + **void Break()** : Működőképes elem elrontására szolgáló függvény.
- + **void GetBroken()** : Visszaadja, hogy az adott elem működőképes-e.

3.3.2 Cistern

- **Felelősség**

Amennyi csőből folyik víz ebbe az elembe, annyi pontot oszt a szerelők csapatának. Erről az elemről a játékosok közvetlen más ciszternáakra tudnak lépni.

- **Ősosztályok**

Element

- **Metódusok**

- + **void OnPickUp()** : megpróbál pumpát adni a soron lévő játékos tárolójába.
- + **void Flood()** : Vízzel tölti fel magát, és pontot ad a szerelőknek. A pálya végpontjaként nem áramoltat más elemekbe vizet.
- + **void Manipulate(Manipulator m)** : A paraméterként kapott manipulátorral manipulálja ezt a konkrét elemet.

3.3.3 Element

- **Felelősség**

Player-eket tud felrakni, illetve eltávolítani magáról, és számon is tartja a rajta tartózkodókat. Tárolja a bemenet/kimenetét, valamint a szomszédjait is, ezeket tudja változtatni, illetve lekérdezni. Továbbá tárolja, hogy van-e víz benne.

- **Attribútumok**

- - **flooded: boolean** : Értéke igaz, ha víz van benne (jelenleg vizet tárol).
- # **pickUpAble: boolean** : Értéke igaz, ha az adott elem felvehető, különben hamis.
- - **players: Player[0..6]** : Azon játékosok halmaza, amelyek az adott elemen tartózkodnak

- - **nbs: Element[1..*]** : A vele szomszédos elemekre hivatkozik.
- - **input: Element[0..1]**: A bemenő szomszédra hivatkozik, ha van ilyen.
- - **output: Element[0..1]** : A kimenő szomszédra hivatkozik, ha van ilyen.

- **Metódusok**

- + void **AddPlayer(Player p)** : Felhelyez magára egy játékost, ha ez lehetséges a rajta állók száma szerint.
- + void **RemovePlayer(Player p)** : Eltávolítja a számoltartott játékosai közül azt, amelyiket paraméterként kapott.
- + Element **GetNb(Direction d)** : Megadott irányba lévő szomszédos elemét adja vissza, ha van ilyen.
- + void **SetNb(Direction d, Element e)** : Megadott irányú szomszédjának állítja be a paraméterként kapott elemet.
- + void **OnPickup()** : Ha egy elem támogatni akarja azt a funkcionálitást, hogy egy szerelő rajta állva egy part-ot kaphasson a tárolójába, akkor ezt a függvényt kell felülírnia. Alapértelmezett megvalósítása üres törzsű függvény (alapjáraton nem támogatják az elemek ezt a funkcionálitást).
- + void **Flood()** : Absztrakt metódus, amely a víz áramoltatásáért felelős.
- + void **Manipulate(Manipulator m)** : Absztrakt metódus. A paraméterként kapott manipulátorral manipulálja az elemet.
- + boolean **GetPickUpAble()** : Visszaadja, hogy az elem felvehető-e (pickUpAble és nincs benne víz)
- + int **GetNbCnt()** : A szomszédos elemek számát adja vissza.
- + int **GetPlayerCnt()** : A rajta tartózkodó játékosok számát adja vissza.
- + boolean **GetFlooded()** : Igazat ad vissza, ha az víz van benne (vagy víz folyik ki belőle), egyéb esetben hamisat.
- + Element **GetInput()** : Visszaadja a bemeneti elemét.
- + Element **GetOutput()** : Visszaadja a kimeneti elemét.
- + void **SetFlooded(boolean f)** : Az átadott érték szerint állítja be, hogy van-e víz jelenleg benne.
- + void **SetInput(Element e)** : A bemenetét az átadott elemre állítja be.
- + void **SetOutput(Element e)** : A kimenetét az átadott elemre állítja be.

3.3.4 Game

- **Felelősség**

A játékmenetet kezeli (vizsgálja, hogy véget ért-e a játék, valamint a köröket vezérli). Számoltartja és módosítja a csőrendszer felépítését (turn-önként véletlenszerűen pumpákat ront el, round-onként pedig ciszternákkal szomszédos üres helyekre új csöveket helyez). A turn-ök elején a forrásokból áramoltatja a vizet a csövekig, vagy ciszternáig. Pontokat oszt a csapatoknak, és számon is tartja azokat.

- **Attribútumok**

- - currPlayerIdx: int : A soron lévő játékos indexe a „players” gyűjteményben.
- - scoreTechnician: int : A szerelők pontszámát tárolja.
- - scoreSaboteur: int : A szabotőrök pontszámát tárolja.
- - input: Gameinput : A felhasználóval kommunikáló objektum.(Híd a View és a Model között.)
- - players: Player[4..6] : A játékban résztvevő játékosokat tárolja.
- - network: Network : A pálya elemeit tárolja, szortírozza.

- **Metódusok**

- + void **AddTechnicianPoints(int points)** : Ez a metódus ad a szerelőknek pontot.

- **+ void AddSaboteurPoints(int points)** : Ez a metódus ad a szabotőröknek pontot.
- **+ void NextTurn()** : A következő turn indítása (áramoltatja a vizet, pontokat oszt, véletlenszerűen pumpákat ront el, és ha véget ért egy teljes kör (round), akkor új csöveket teremt a ciszternák üres szomszédjai helyén). Ha az egyik csapat elérte a győzelemhez szükséges pontok számát, véget vet a játéknak.
- **+ Player GetCurrPlayer()** : Visszaadja a játékost, aki éppen soron van (akinek turnje van jelenleg).
- **+ boolean IsTechnicianTurn()** : Igazat ad vissza, ha éppen szerelő jön az adott körben.
- **+ GameInput GetInput()** : Visszaadja a felhasználóval kommunikáló objektumot.
- **+ Network GetNetwork** : Visszaadja a pálya elemeit számontartó objektumot.

3.3.5 GameInput

- **Felelősség**

A felhasználót és a játéket köti össze. Irányokat / célpontokat tud bekérni a felhasználótól. Játékosmozgatást, cső- és pumpalehelyezést, illetve csőfelvezést kezdeményezhet vele. Pumpát is tud adni a jelenlegi játékosnak, amennyiben egy szerelő, ciszernán áll és üres a tárolója.

- **Metódusok**

- + void **MoveCurrPlayer(Direction d)** : d irányba próbálja mozgatni az épp soron lévő játékost, arról az elemről, amelyiken éppen tartózkodik.
- + void **TryCurrElemManipulation()** : akkor hívandó függvény, amikor éppen azt az elemet szeretné manipulálni / interakcióba lépni vele (csövet lyukasztani / foltozni, pumpát javítani / átállítani vagy a következő ciszternára ugrani) a soron lévő játékos, amelyiken éppen áll.
- + void **TryPartRelocation(Direction d)** : akkor hívandú függvény, amikor a soron lévő játékos megpróbál egy tőle d irányba lévő szomszédos csövet felvenni, és a tárolójában eltárolni.
- + void **TryPartPlacement(Direction d)** : akkor hívandú függvény, amikor a soron lévő játékos megpróbálja a tárolt part-ját tőle d irányba lehelyezni.
- + void **Pickup()** : akkor hívandó függvény, amikor egy játékos egy part-ot próbálja a tárolójába tenni (nem a játéktérről, hanem közvetlenül odaadva). Ez csak akkor lesz sikeres, ha a játékos szerelő, üres a tárolója, és az elem amin áll, támogat ilyen funkcionálitást.
- + Direction[2] **GetNewPumpDirections()** : pumpa átállításához visszaadja a soron lévő játékos által bevitt irányokat.

3.3.6 Manipulator

- **Felelősség**

Absztrakt osztály, amely a Visitor tervezési mintát valósítja meg a leszármazottaival együtt. A leszármazottai egy játékos típusnak készítik el a „gazda” elem manipulálását megvalósító viselkedést minden elem esetére (pumpák, csövek, források, ciszternák).

A szerelők konkrét manipulátora például definiálja, hogy mit tud tenni egy szerelő, ha az előbb említett elemeken állva próbál interaktálni. A függvényei gyakran a kört is léptetik.

- **Metódusok**

- + void **Manipulate(Pipe p)** : Absztrakt metódus a játékosok cső manipulálásának leírására.
- + void **Manipulate(Pump p)** : Átállítja az átadott pumpát (GameInput-ot használja a bemenetért). Ezután véget ér a játékos köre (turn).
- + void **Manipulate(Cistern c)** : Átlépteti a jelenlegi játékost a következő ciszternára. Ezzel nem ér véget a játékos köre (turn).
- + void **Manipulate(Source s)** : Üres törzsű függvény, jelenleg a játékosok nem tesznek semmit a forráson állva. (A jövőbeli bővíthetőségre fenntartva, illetve a Dynamic Dispatch hibátlan működése miatt kell, hogy ne kelljen Type-checkingelni a hívóoldalon.)

3.3.7 ManipulatorSaboteur

- **Felelősség**

A szabotörök konkrét manipulátora, ami definiálja, hogy hogyan manipulálhatják az összes lehetséges „gazda” elemüket.

A csövek kilyukasztásának viselkedését valósítja meg. A többi viselkedés megfelel az űsbélivel.

- **Ősosztályok**

Manipulator

- **Metódusok**

- + void **Manipulate(Pipe p)** : Kilyukasztja az átadott p csövet. Utána pedig véget ér a jelenlegi játékos köre.

3.3.8 ManipulatorTechnician

- **Felelősség**

A szerelők konkrét manipulátora, ami definiálja, hogy hogyan manipulálhatják az összes lehetséges „gazda” elemüket.

A csövek javításának viselkedését valósítja meg, illetve a pumpák megjavításának viselkedésével helyettesíti a pumpaátállítást, ha az adott pumpa rossz.

A többi viselkedés megfelel az űsbélivel.

- **Ősosztályok**

Manipulator

- **Metódusok**

- + void **Manipulate(Pipe p)** : Megjavítja az átadott p csövet. Utána pedig véget ér a jelenlegi játékos köre.
- + void **Manipulate(Pump p)** : Megjavítja az átadott p pumpát, ha az elromlott, különben pedig átállítja (ez esetben meghívja az űsbéli megvalósítását a függvénynek).

3.3.9 Network

- **Felelősség**

Tárolja a pálya elemeit. Csövek és pumpák adhatók hozzá, a csöveket pedig el is lehet távolítani róla. Vizsgálja, hogy ezek a műveletek lehetségesek-e egyáltalán.

A benne tárolt elemek azok, amik ténylegesen meg is lesznek jelenítve a pályán.

Továbbá ez az osztály indítja el a vízfolyást forrásokból.

- **Attribútumok**

- - **elements: Element[4..*]** : A pálya összes elemét tárolja.
- - **cisterns: Cistern[1..*]** : Külön csoportosítja minden elem közül a ciszternákat ez a gyűjtemény.
- - **sources: Source[1..*]** : Külön csoportosítja minden elem közül a forrásokat ez a gyűjtemény.

- - **pumps: Pump[0..*]** : Külön csoportosítja minden elem közül a pumpákat ez a gyűjtemény.
- **Metódusok**
 - + **boolean RemoveElem(Element e)** : Megkísérli eltávolítani az átadott elemet a pályáról. Ha ennek semmi akadálya nem volt (pl. nem esne komponensekre a pálya, mint gráf), akkor igazzal tér vissza. Ellenkező esetben nem módosít semmi a pályán.
 - + **boolean AddPipe(Pipe p)** : Az átadott csövet megkísérli hozzáadni a pályához. Ennek sikerességét adja vissza.
 - + **boolean AddPump(Pump p, Element e)** : Az átadott pumpával kísérli meg felülírni a másik átadott elemet a pályán. Ennek sikerességét adja vissza.
 - + **void Flood()** : Felapasztja az összes vizet a pályáról, majd minden forrásból elindítja a vizet, aminek következtében el lesz árasztva vízzel a pálya, és pontokat fognak kapni a csapatok.
 - + **Element[4..*] GetElements()** : Visszaadja a pálya minden elemét.
 - + **Pump[0..*] GetPumps()** : Visszaadja a pályán lévő pumpákat.
 - + **Cistern[1..*] GetCisterns()** : Visszaadja a pályán lévő ciszternákat.

3.3.10 Pipe

- **Felelősség**

Olyan elromolható, felvehető elem, amelyen legfeljebb egy játékos tartózkodhat.

Ki lehet lyukasztani, a lyukas csövet pedig meg is lehet javítani. Ha a cső nem lyukas akkor a kimenetéhez vizet tud juttatni.

Ha lyukas, vagy a kimenete üres, akkor a szabotőrök pontot kapnak, amikor víz érkezik belé.

- **Ősosztályok**

Element → Breakable

- **Metódusok**

- + void **AddPlayer(Player p)** : Felhelyezi az átadott játékost magára, ha nem áll rajta már más valaki.
- + void **Manipulate(Manipulator m)** : Az átvett manipulátorral manipulálja ezt a konkrét elemet.
- + void **Flood()** : Vízzel tölti meg magát. Ha lyukas a cső vagy nincs output-ja akkor pontot kapnak a szabotőrök, különben pedig hívja tovább az output-ján a Flood() függvényt (ereszti tovább a vizet).

3.3.11 Player (Saboteur)

- **Felelősség**

A Player osztály a szabotőr osztállyal ekvivalens, tehát a szabotőrök csapatának játékosai Player típusúak. Eltárolja a játékos nevét, az elemet, amelyen éppen áll, és a manipulátorát. Tud mozogni irányokba, és manipulálni tudja a „gazda” elemét.

- **Attribútumok**

- - currElem: Element : Az adott elem, amelyen a játékos tartózkodik
- - name: string : A játékos neve, azonosítója
- - manipulator: Manipulator : A játékos által használt manipulator objektum a „gazda” elemmel való interakciók lekezeléséhez.

- **Metódusok**

- + void **MoveTo(Element e)** : Az átadott elemre helyezi a játékost, ha ez lehetséges.
- + void **StepOnto(Direction d)** : A játékost a megadott irányban lévő elemre lépteti, ha ez lehetséges.
- + void **ManipulateCurrElem()** : Manipulálja azt az elemet, amelyen éppen tartózkodik. A manipulátora határozza meg, hogy mely „gazda” elem típus esetén mit tesz.
- + Element **GetCurrElem()** : Visszaadja az elemet, amelyen a játékos éppen tartózkodik.

3.3.12 Pump

- **Felelősség**

Olyan elromolható, felvehető elem, amely vizet képes átereszteni a bemenetként beállított csövéről a kimenetként beállított csövébe. Pontot oszt a szabotöröknek, ha nincsen kimenete, amikor vizet ereszte át.

- **Ósosztályok**

Element → Breakable

- **Metódusok**

- + **void Flood()** : Ha a pumpa el van romolva, nem tesz semmit. Ha nincs elromolva akkor vízzel tölti fel magát. Ha nincsen outputja, akkor a szabotörök pontot kapnak, de ha van outputja, akkor az outputra hívja tovább a Flood() függvényt.
- + **void Manipulate(Manipulate m)** : Az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.13 Source

- **Felelősség**

Olyan elem, amely vizet juttat a vele szomszédos elemekbe, és soha sincsen bemenete.

- **Ósosztályok**

Element

- **Metódusok**

- + **void Flood()** : Vízzel tölti fel magát, és minden nem lyukas szomszédjára hívja tovább a Flood() függvényt, azaz ereszti tovább a vizet. Ha nem tudja ezt megtenni az adott szomszédja felé, mert lyukas, vagy nincs arra szomszédja, akkor pontot ad a szabotöröknek.
- + **void Manipulate(Manipulator m)** : Az átvett manipulátorral manipulálja ezt a konkrét elemet.

3.3.14 Technician

- **Felelősség**

Olyan játékos, aki csövek lyukasztása helyett javítani tudja őket, illetve pumpákat is. Emellett képes vagy egy csövet, vagy egy pumpát tárolni magánál, ami el is tud helyezni a megfelelő feltételek fennállása esetén.

- **Ósosztályok**

Player

- **Attribútumok**

- - **part: Breakable[0..1]** : Az az elem, amelyet a szerelő felvett a pályáról.

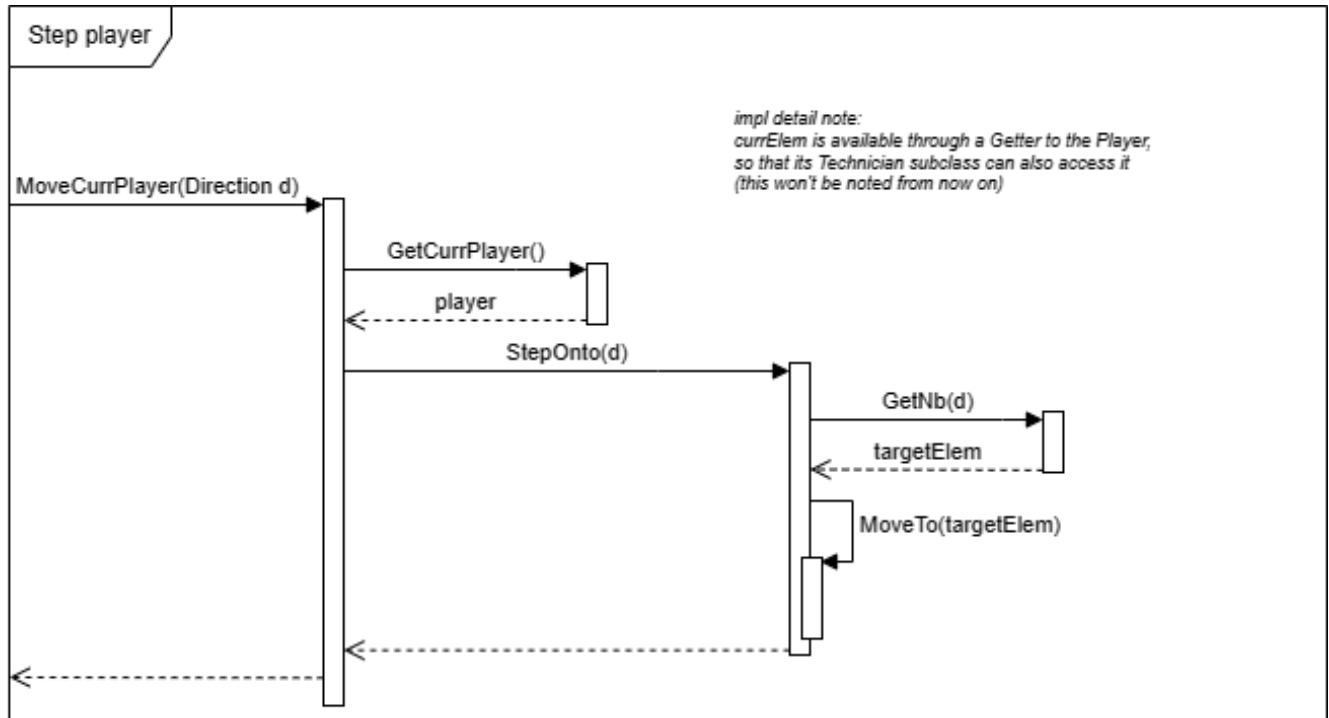
- **Metódusok**

- + **void PickUpPart()** : Megkísérel felvenni egy part-ot a jelenlegi elemtől.
- + **void PickUpPart(Direction d)** : Megkísérli felvenni a d irányban lévő part-ot.

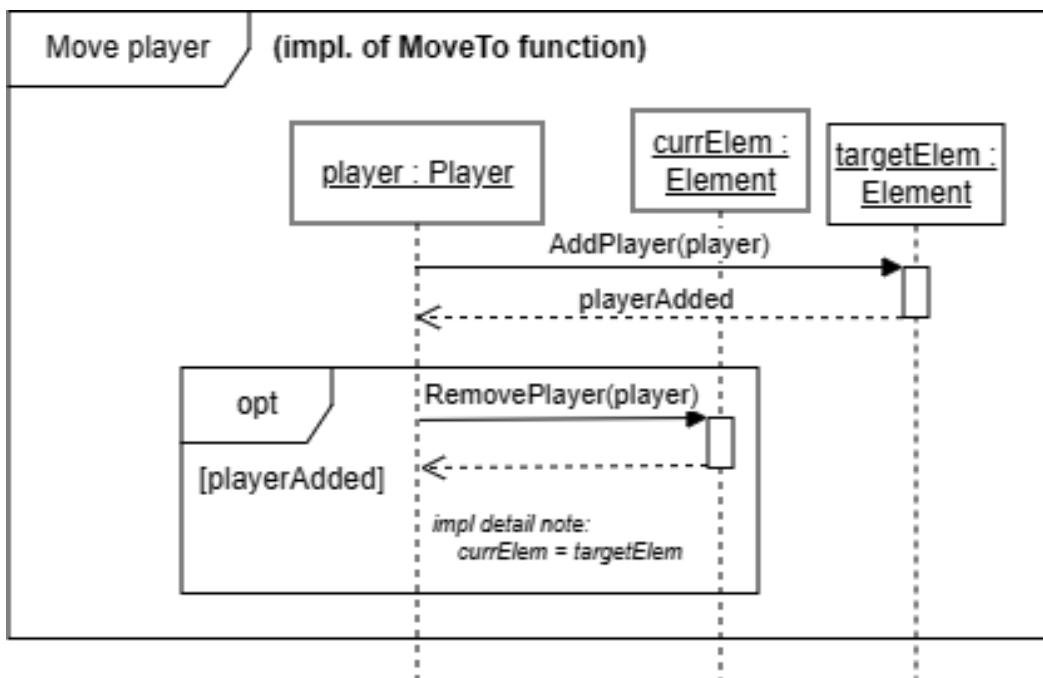
- + **boolean PlacePart(Direction d)** : Megkísérli elhelyezni a tárolt part-ját d irányba. A művelet sikerességevel tér vissza.
- + **Breakable GetPart()** : Visszaadja azt a Breakable-t („part” attribútumot), ami a szerelőnél van.
- + **void SetPart(Breakable b)** : Az átadott Breakable-re állítja a „part” attribútumot.

3.4 Szekvencia diagramok

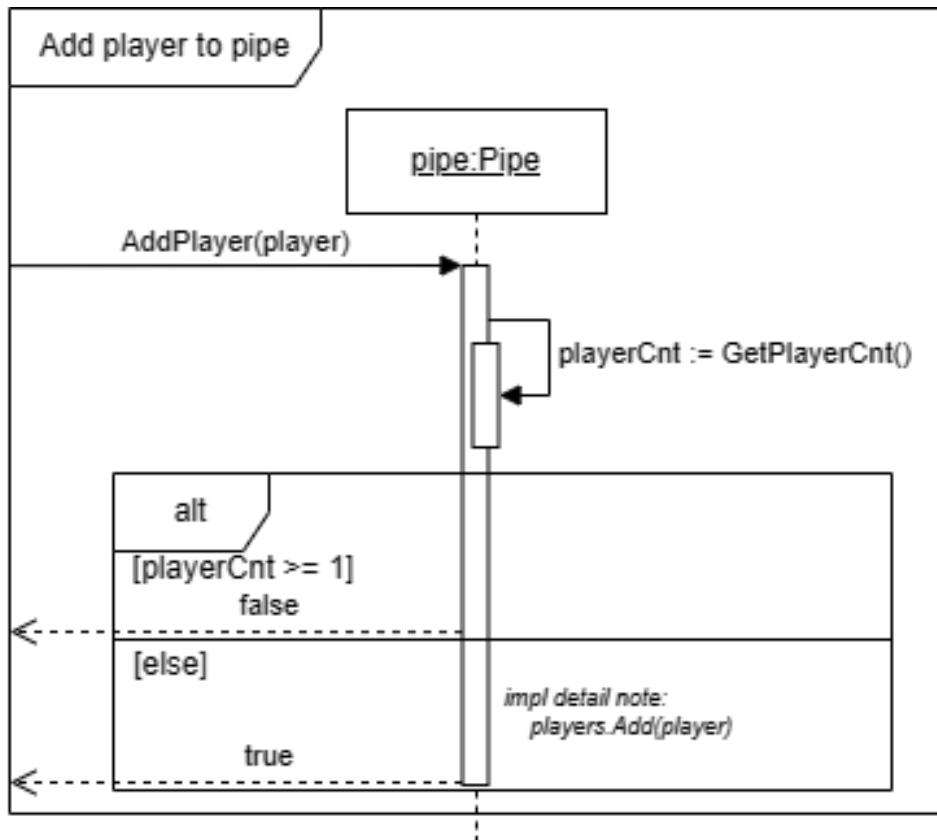
3.4.1 Step Player



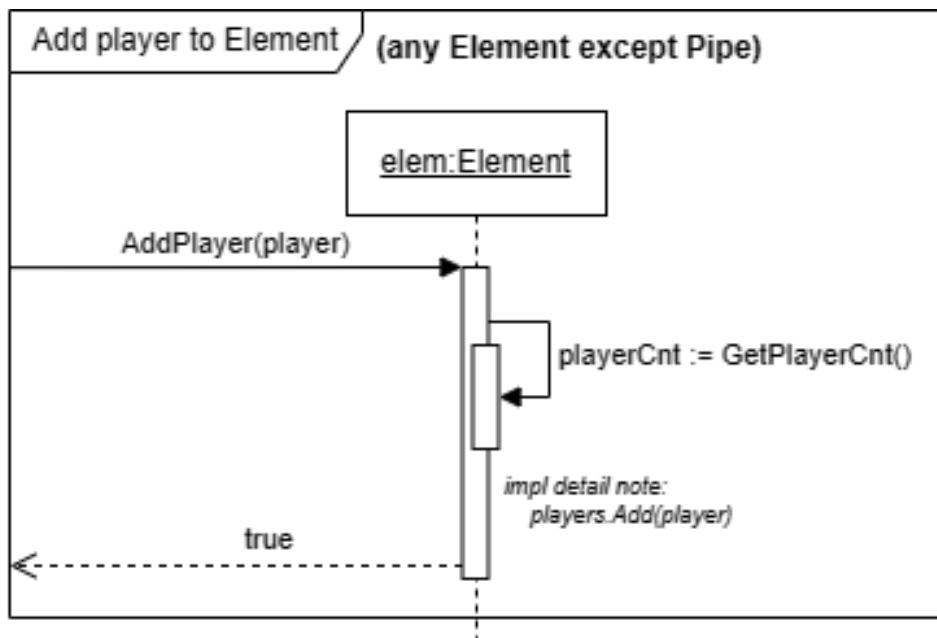
3.4.2 Move Player



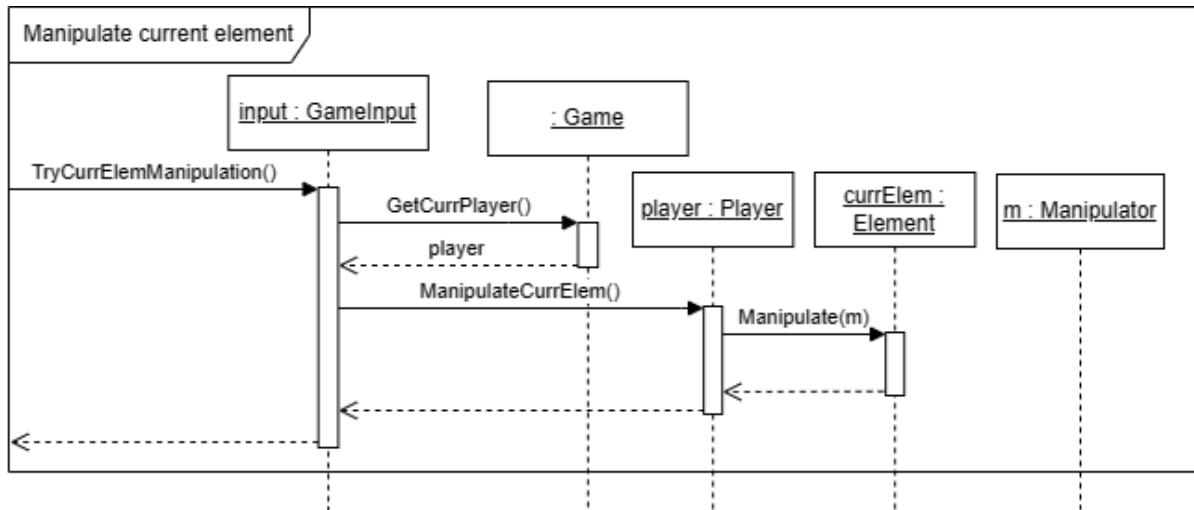
3.4.3 Add Player to Pipe



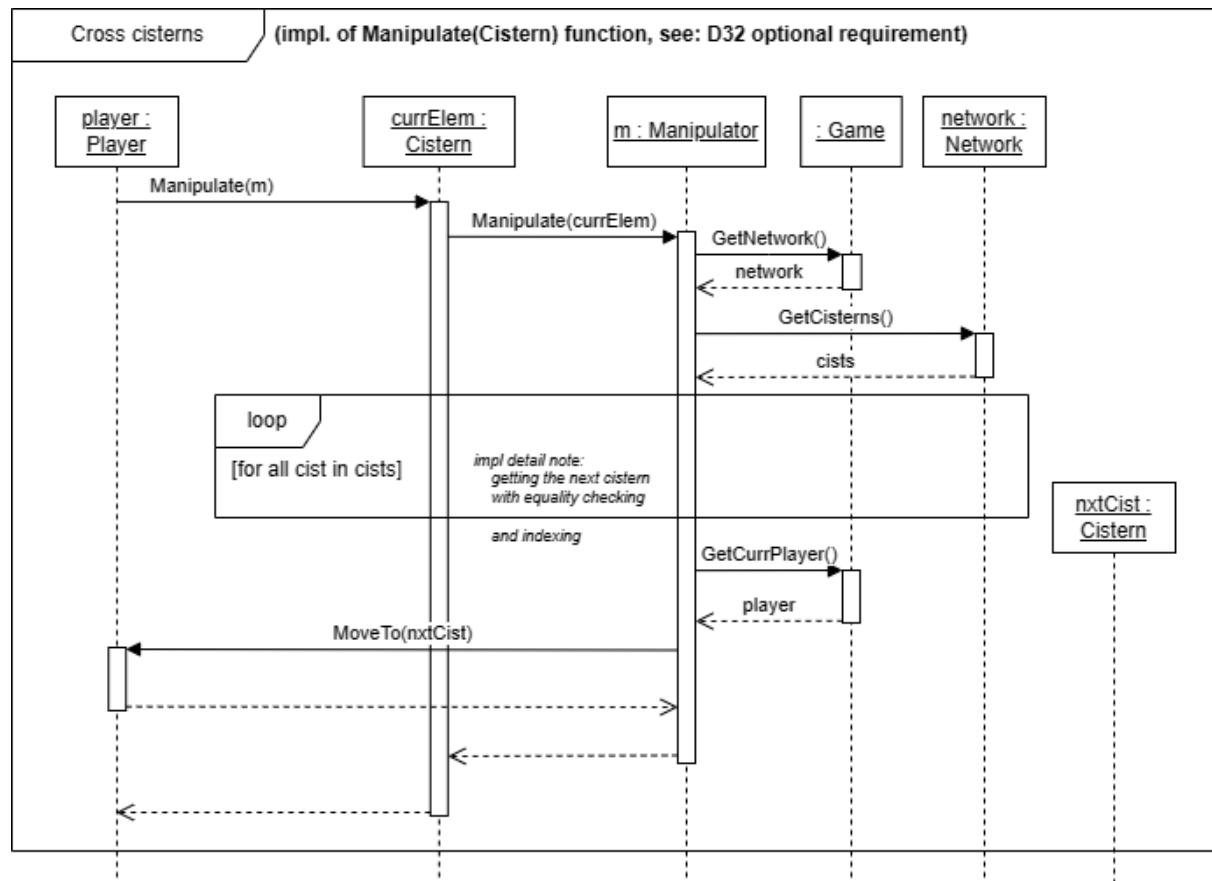
3.4.4 Add Player to Element



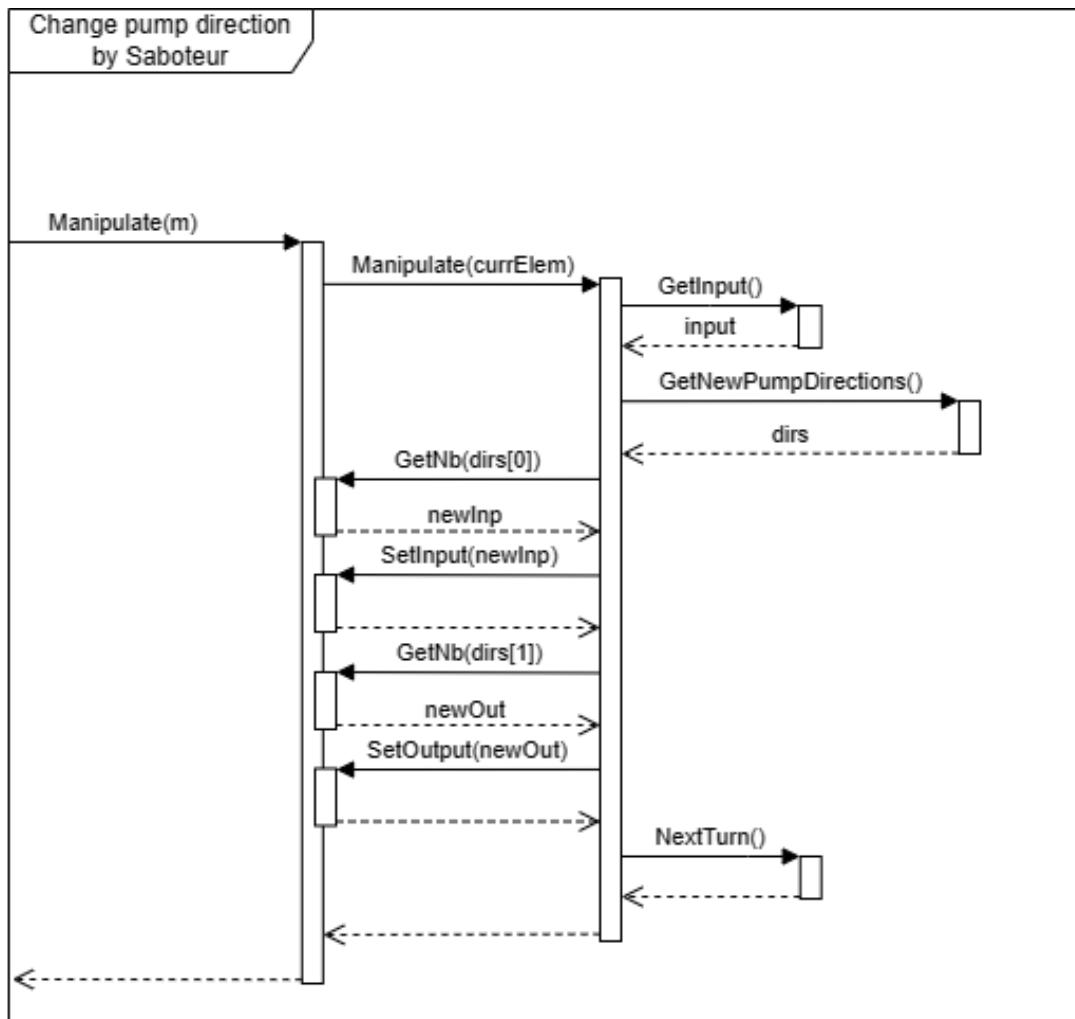
3.4.5 Manipulate Current Element



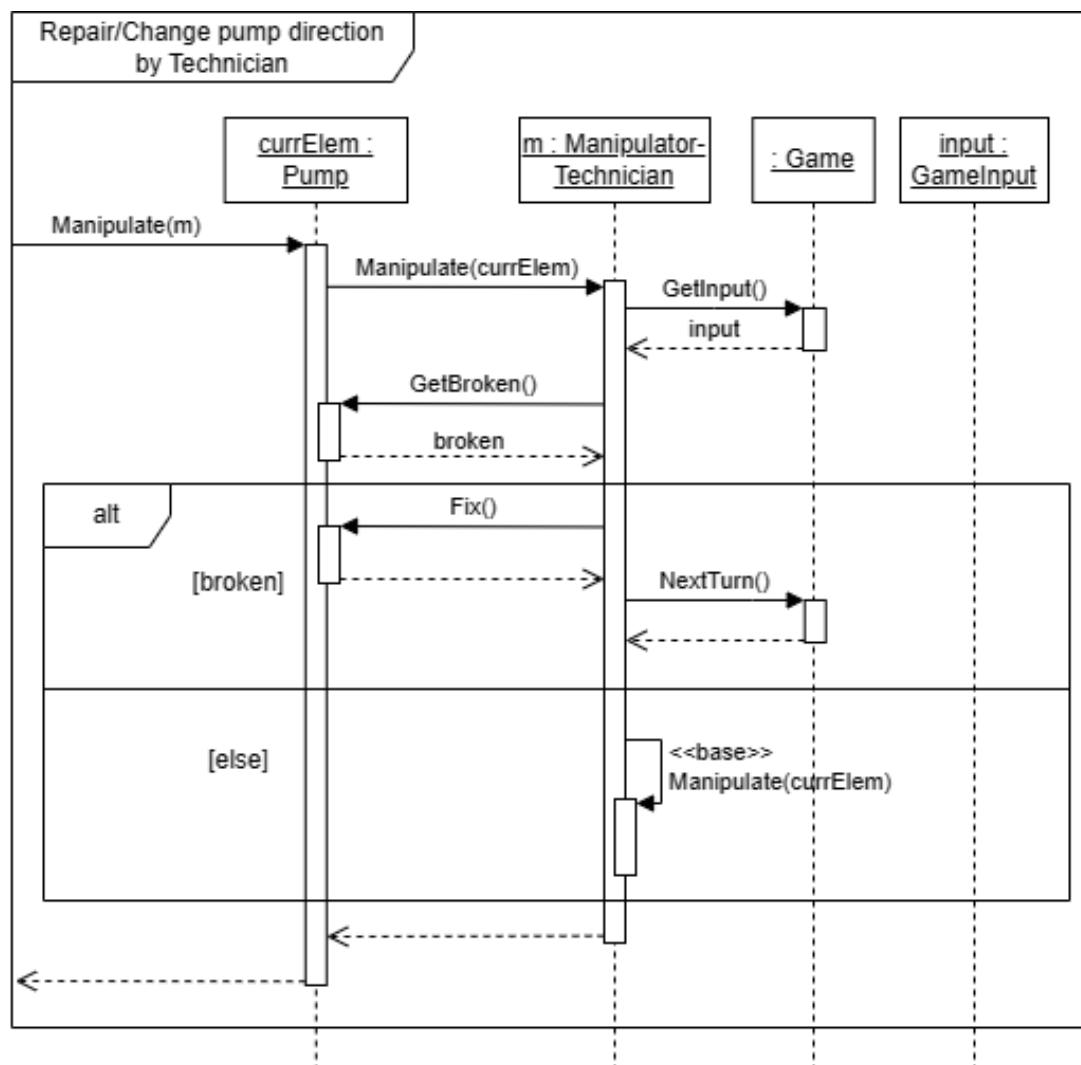
3.4.6 Cross Cisterns



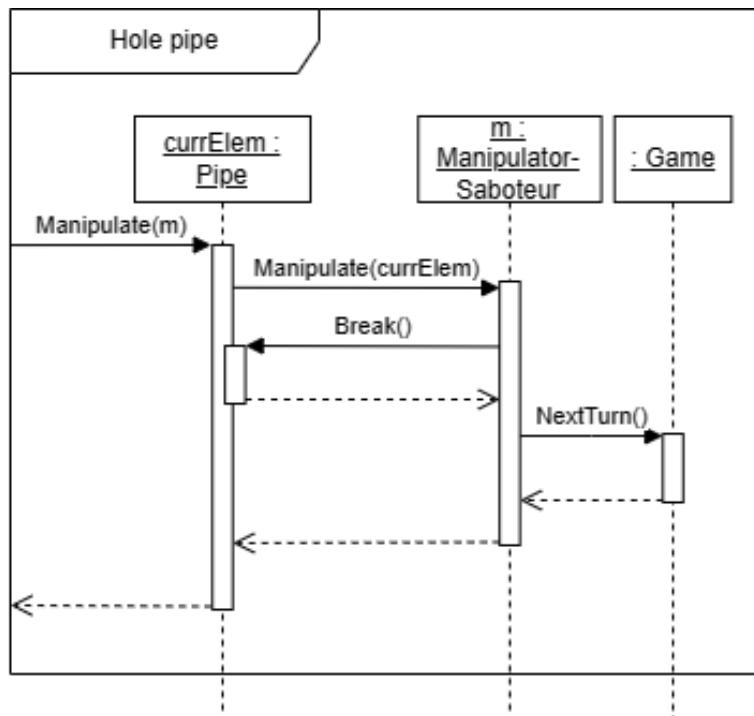
3.4.7 Change Pump Direction (by Saboteur)



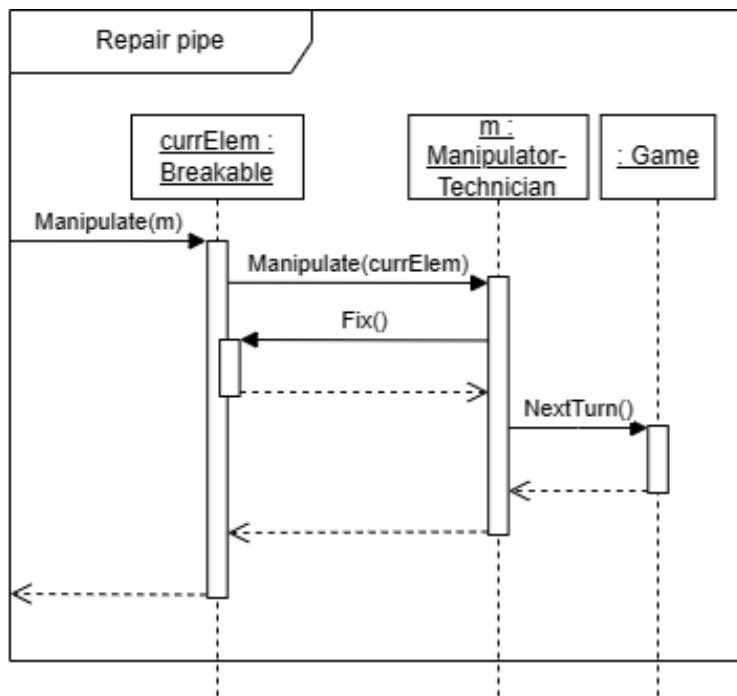
3.4.8 Repair or Change Pump Direction (by Technician)



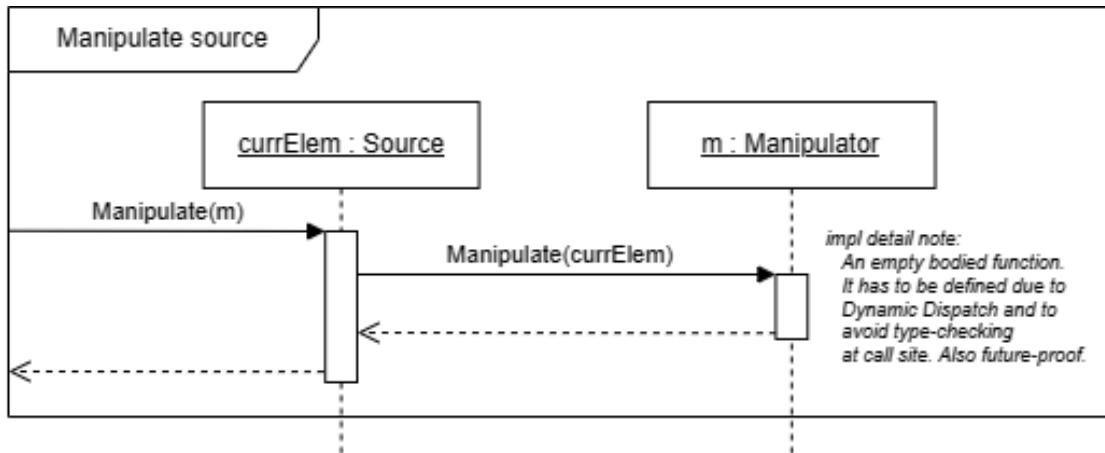
3.4.9 Hole Pipe



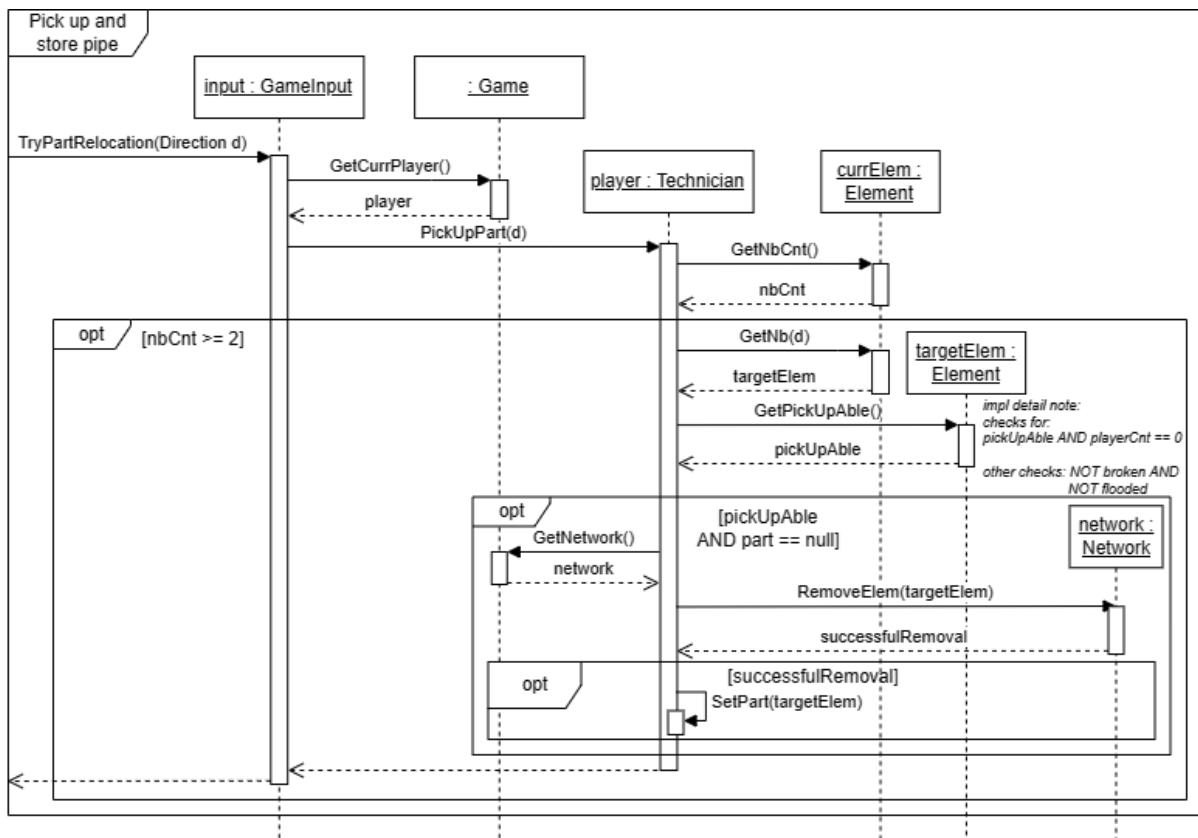
3.4.10 Repair Pipe



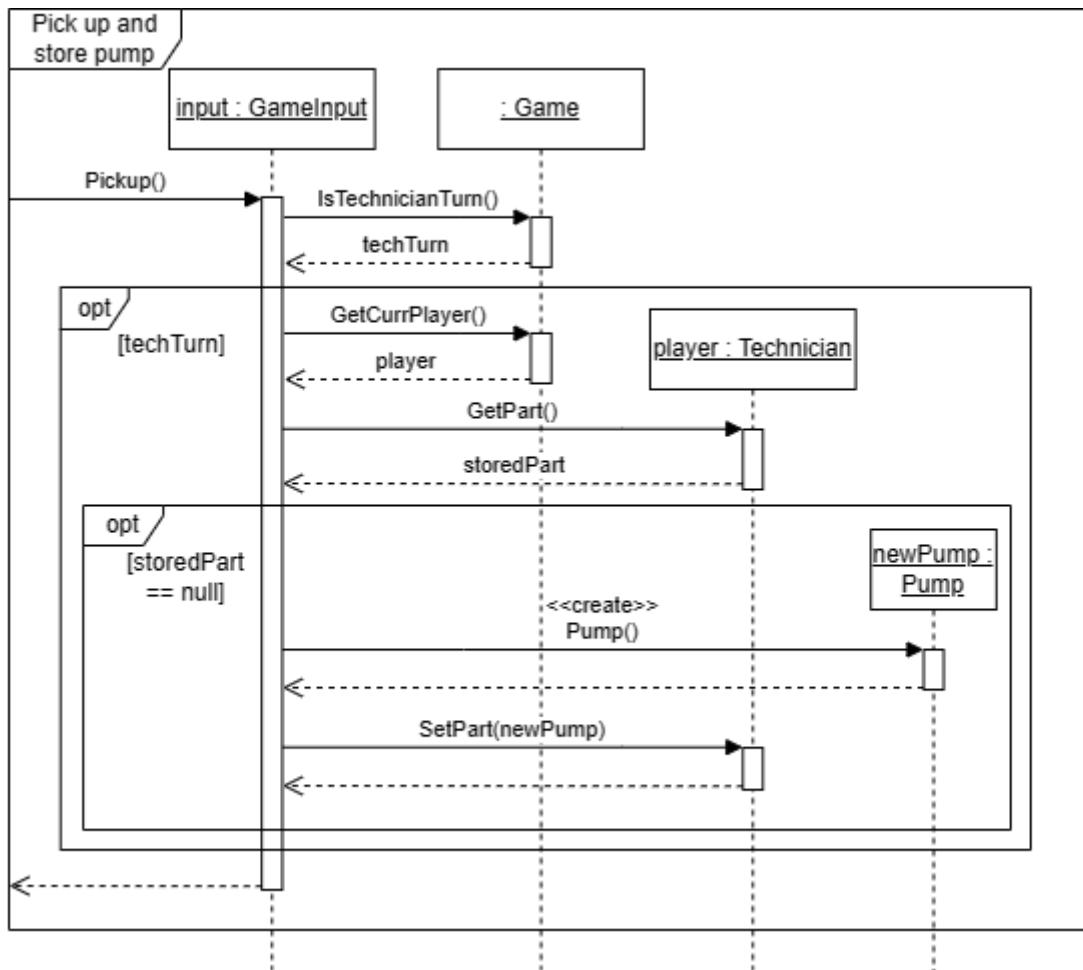
3.4.11 Manipulate Source



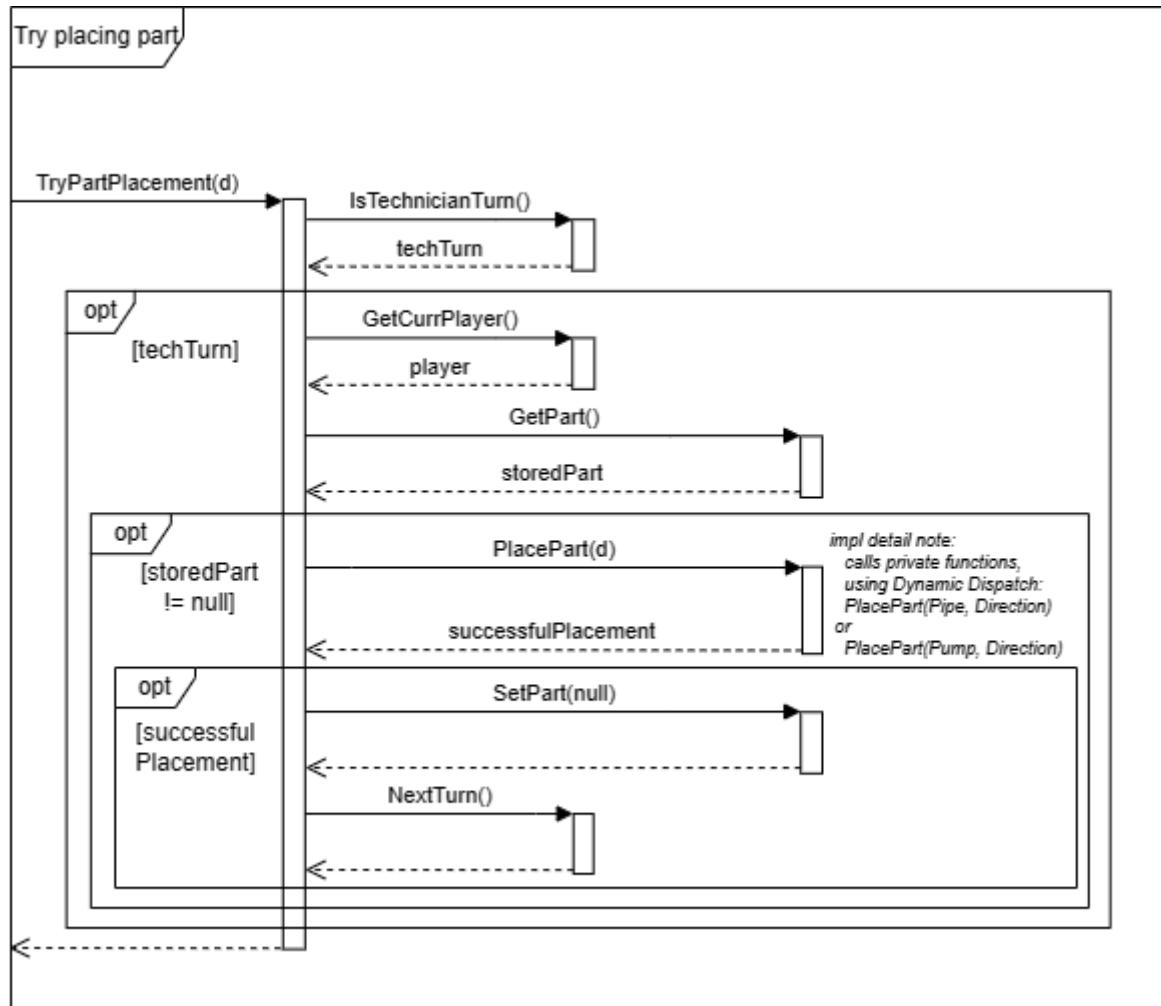
3.4.12 Pick Up and Store Pipe



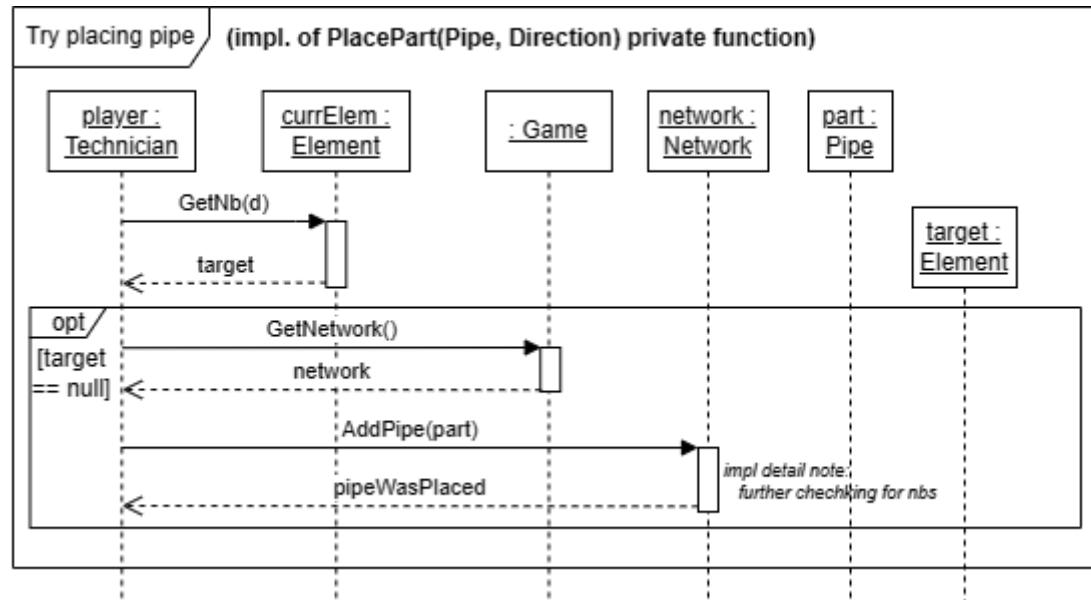
3.4.13 Pick Up and Store Pump



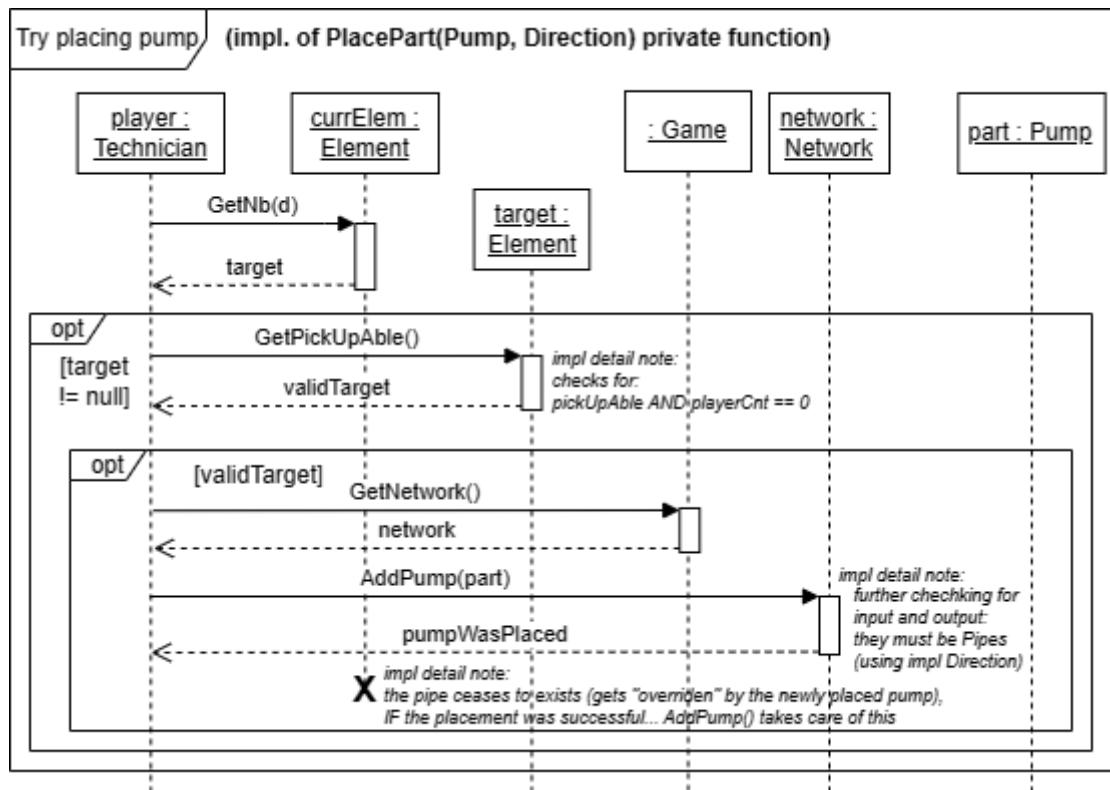
3.4.14 Try Placing Part



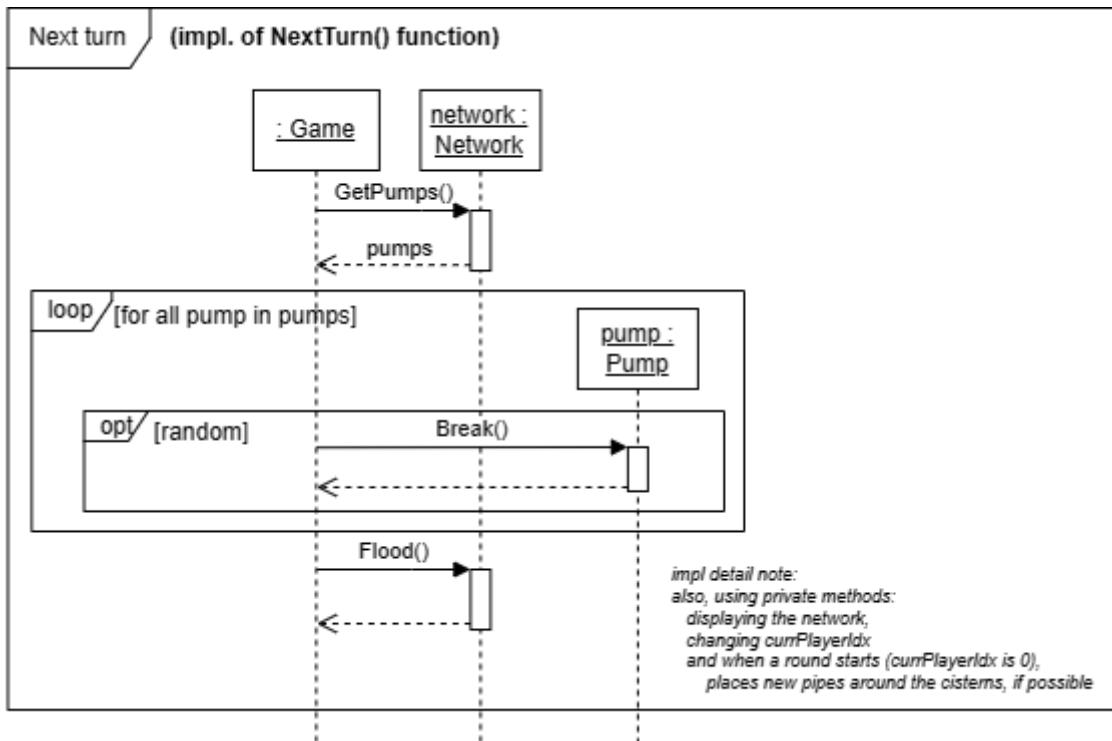
3.4.15 Try Placing Pipe



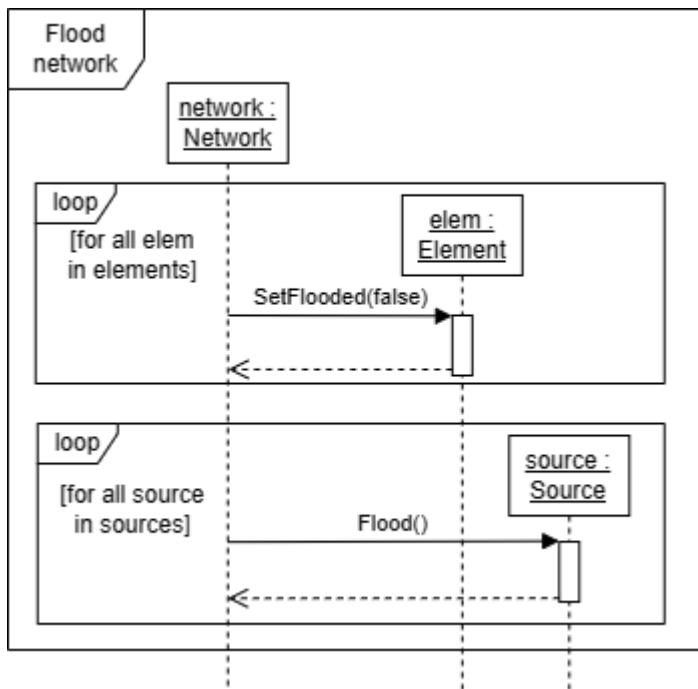
3.4.16 Try Placing Pump



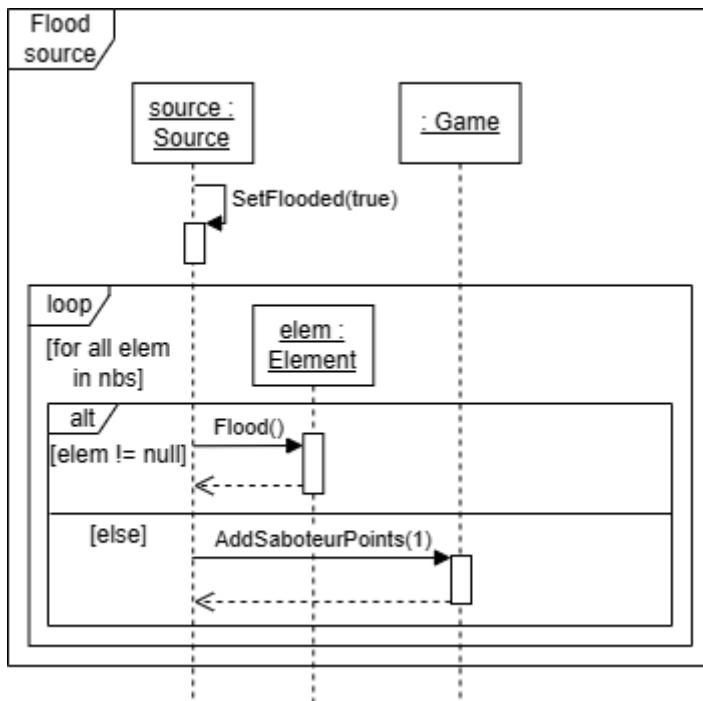
3.4.17 Next Turn



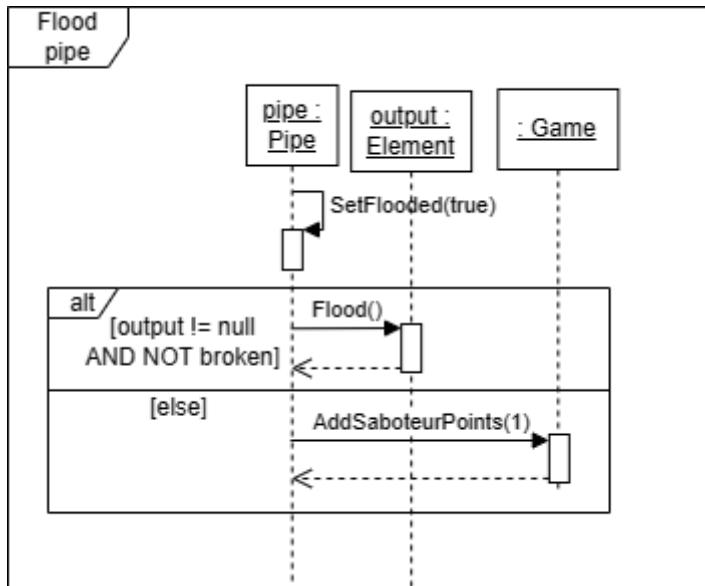
3.4.18 Flood Network



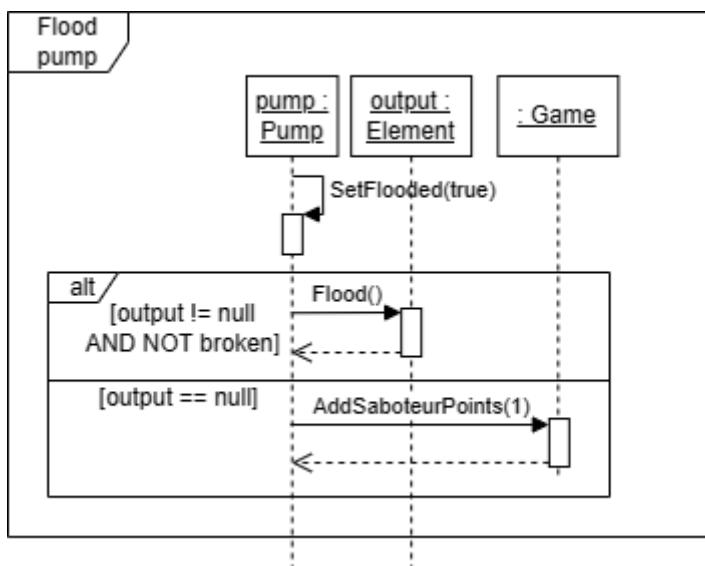
3.4.19 Flood Source



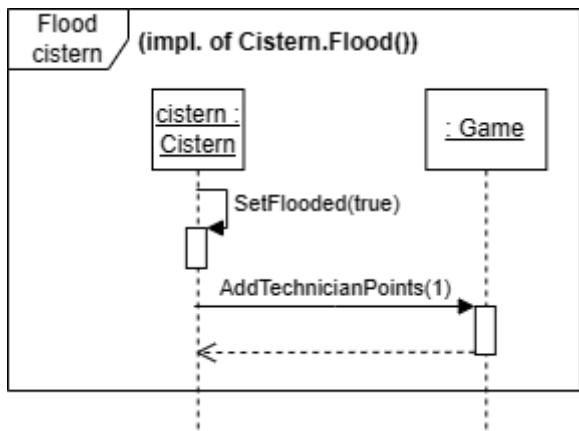
3.4.20 Flood Pipe



3.4.21 Flood Pump

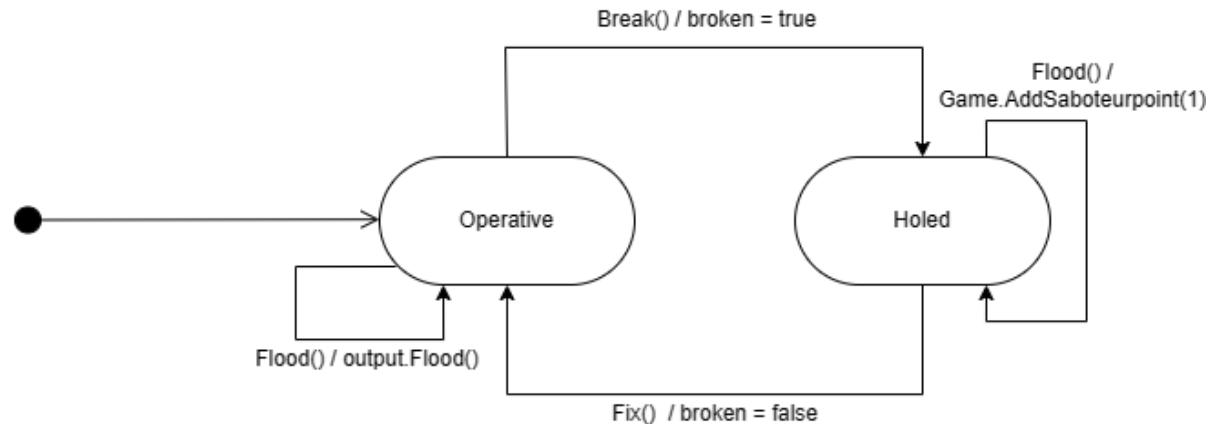


3.4.22 Flood Cistern

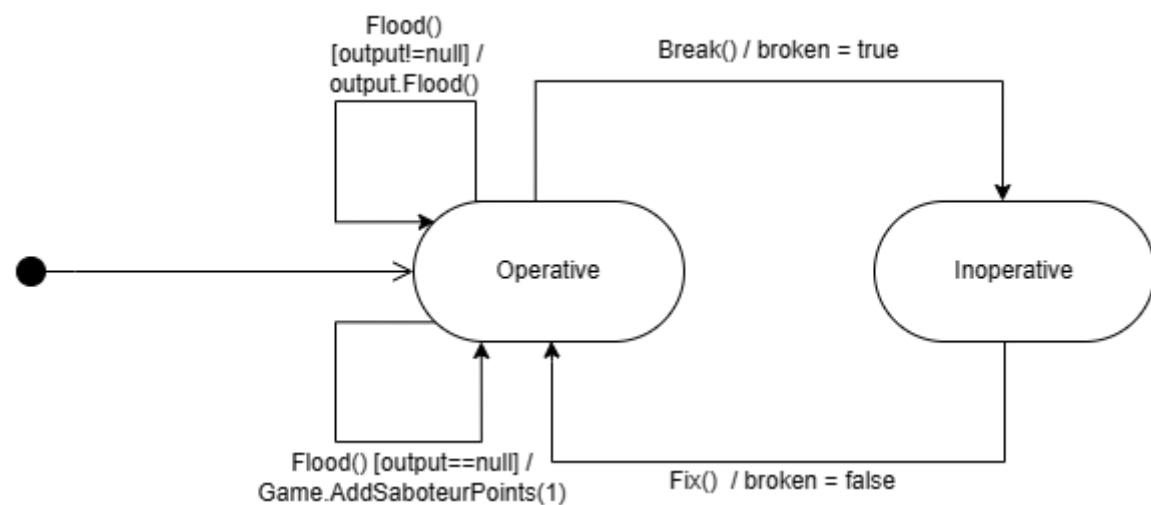


3.5 State-chartok

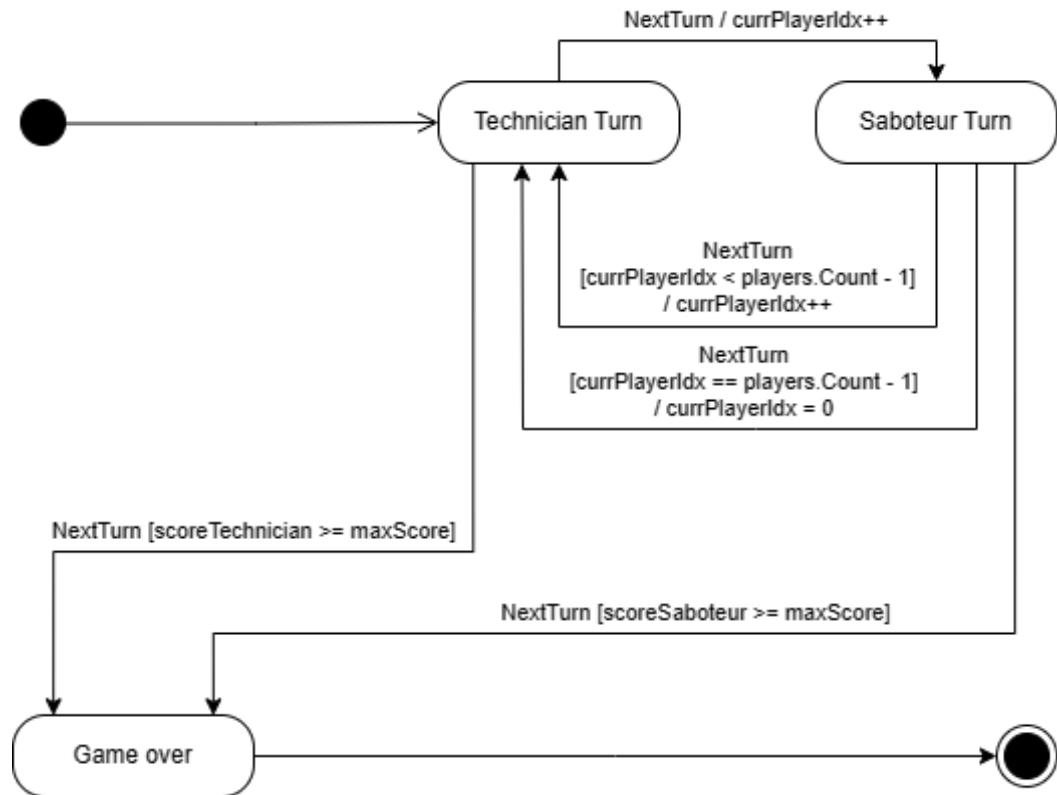
3.5.1 Pipe States



3.5.2 Pump States



3.5.3 Játékos köreinek (turn) váltakozása



3.6 Napló

Kezdet	Időtartam	Részttvevők	Leírás
2023.03.22. 19:00	0,5 óra	Mizser Váradi Tepliczky Fekete Sasvári	<p>Értekezlet. Döntés:</p> <ul style="list-style-type: none"> • Mizser és Sasvári javítja az osztálydiagramot és a szekvenciadiagramokat • Fekete, Tepliczky és Sasvári elkészíteti a 2 hiányzó szekvenciadiagramot • Váradi javítja a dokumentum formai és tartalmi hiányosságait <hr/> <p>Diagram javítások Határidő: 2023.03.25. 22:00. Elkészült: 2023.03.25. 20:00</p> <p>Kész dokumentum (formázással) Határidő: 2023.03.27. 0:00 Elkészült: 2023.03.26 21:00</p>
2023.03.23. 18:00	3 óra	Mizser Sasvári	<p>Osztálydiagram és szekvenciadiagramok hibáinak feltárása:</p> <ul style="list-style-type: none"> • cső életvonalának megszakadása amikor felülíródik a lerakott pumpával • "Repair/Change pump direction by Technician"-ben a felesleges GetPumpDirections hívás törlése • GetCurrElem getter hozzáadása Playerhez, ennek jelölése szekvencián megjegyzéssel • MoveTo(targetElem) felvétele osztálydiagramra • GetPickUpAble függvény leírásának megváltoztatása
2023.03.23 20:00	2 óra	Mizser	<p>Az előzőek tényleges módosítása a diagramokon. További javítások:</p> <ul style="list-style-type: none"> • „osztálydiagram magyarázatok” rész megírása • nem egyértelmű szekvencia diagramokra az illusztrált függvény nevének kiírása • konzisztens függvénynevek
2023.03.25. 18:30	1,5 óra	Tepliczky, Fekete, Sasvári	Pipe States és Pump States állapotdiagramok elkészítése. Korábbi javítások áttekintése.
2023.03.26. 14:00	1 óra	Váradi	<p>A dokumentum formai és tartalmi javítása:</p> <ul style="list-style-type: none"> • A hiányzó oldalszámok pótlása • Objektumkatalógus pontosítása

2023.03.26. 17:00	1 óra	Váradi	A dokumentum tartalmi javítása: • Pontosítás és korrekció az Objektumkatalógus szövegében és az osztályleírásokban.
2023.03.26. 19:00	2 óra	Váradi	A dokumentum tartalmi és formai javítása: • További apróbb korrekciók az osztályok leírásában. • Szekvenciadiagramok frissítése. • Állapotdiagramok frissítése. • Dokumentum tördelése és formázása.

5. Szkeleton tervezése

5 – *runtime_error*

Konzulens:
Dobos-Kovács Mihály

Csapattagok

Mizser Ádám Zoltán	SHKGZW	mizser.adam@gmail.com
Tepliczky Olivér	WB6LC5	tepliczkyo@edu.bme.hu
Fekete Álmos Valér	KR5WPC	feketealmos0@gmail.com
Váradi Kristóf	BP17IB	kristofvaradi@edu.bme.hu
Sasvári Szabolcs Attila	TWOZG6	szabolcs.attila.sasvari@edu.bme.hu

2023.04.03.

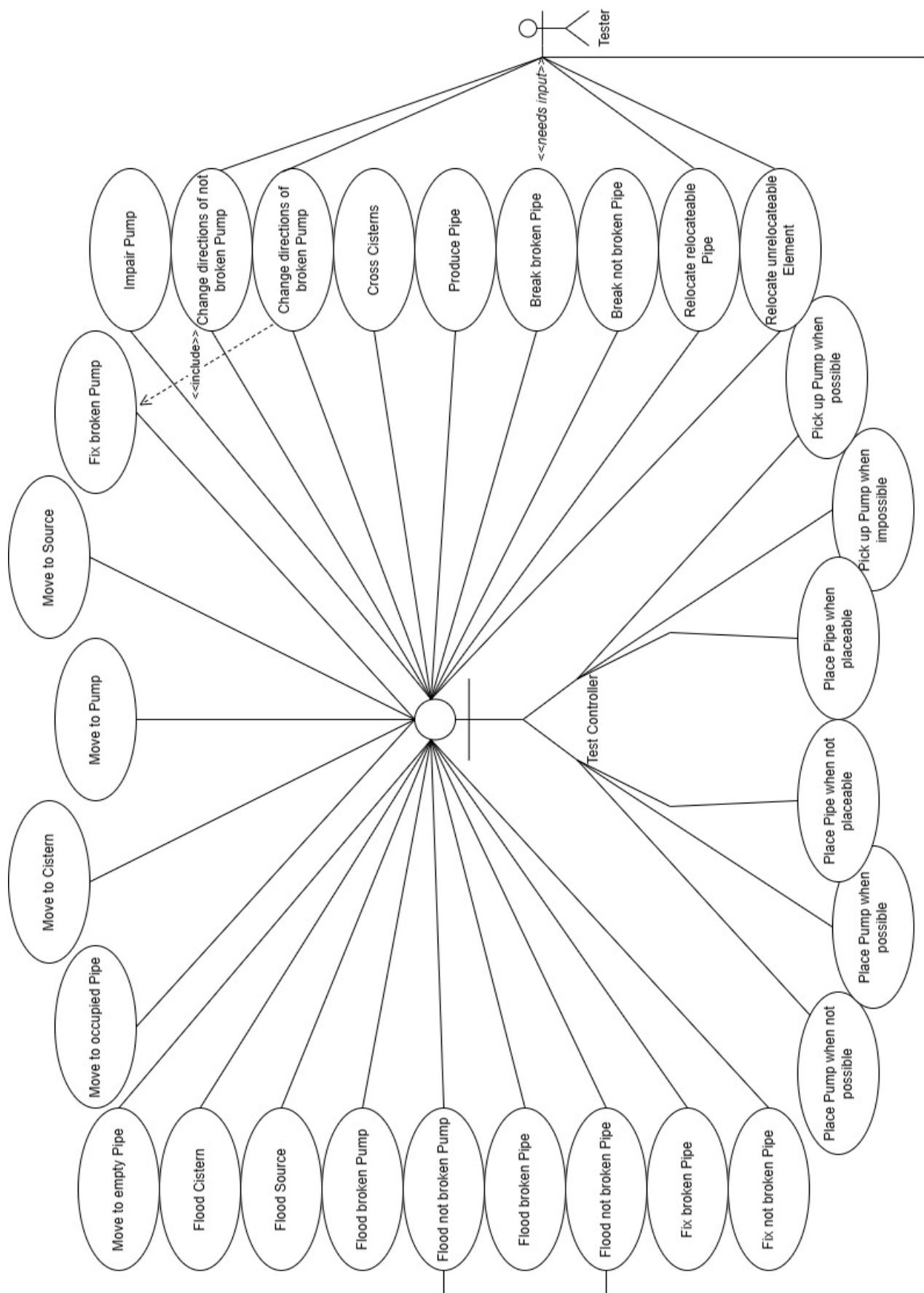
5. Szkeleton tervezése

5.1 A szkeleton modell valóságos use-case-ai

Use-case-ek listája (részleteik 5.1.2-nél):

1. Move to empty Pipe
2. Move to occupied Pipe
3. Move to Pump
4. Move to Cistern
5. Move to Source
6. Fix broken Pump
7. Change directions of not broken Pump
8. Change directions of broken Pump
9. Cross Cisterns
10. Break not broken Pipe
11. Break broken Pipe
12. Fix not broken Pipe
13. Fix broken Pipe
14. Relocate relocatable Pipe
15. Relocate unrelodable Element
16. Pick up Pump when possible
17. Pick up Pump when impossible
18. Place Pipe when placeable
19. Place Pipe when not placeable
20. Place Pump when placeable
21. Place Pump when not placeable
22. Flood not broken Pipe
23. Flood broken Pipe
24. Flood not broken Pump
25. Flood broken Pump
26. Flood Source
27. Flood Cistern
28. Produce Pipe
29. Impair Pump

5.1.1 Use-case diagram



5.1.2 Use-case leírások

Use-case neve	Move to empty Pipe
Rövid leírás	Egy játékost egy szomszédos, üres csőre léptet.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy játékost megkísérel egy csőről egy szomszédos, üres csőre mozgatni.</p> <p>2. A játékost eltávolítja a jelenlegi csövéről.</p> <p>3. Ráteszi a játékost a másik csőre.</p>
Kapcsolatok:	an. req.: D13, D30 an. u-c.: Move Player an. seq.: 3.4.2, 3.4.3 sk. seq.: 5.3.1 sk. com.: 5.4.1

Use-case neve	Move to occupied Pipe
Rövid leírás	Egy játékost megkísérel olyan csőre léptetni, amin már állnak.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy játékost egy csőről egy szomszédos, nem üres csőre akarja mozgatni.</p> <p>2. A cső nem üres, így a játékos a jelenlegi csövön marad.</p>
Kapcsolatok:	an. req.: D13, D30 an. u-c.: Move Player an. seq.: 3.4.2, 3.4.3 sk. seq.: 5.3.2 sk. com.: 5.4.2

Use-case neve	Move to Pump
Rövid leírás	Egy játékost egy pumpára lépett.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy játékost megkísérel egy csőről egy szomszédos pumpára mozgatni.</p> <p>2. A játékost eltávolítja az adott csőről.</p> <p>3. Ráteszi a játékost a pumpára.</p>
Kapcsolatok:	an. req.: D13, D30 an. u-c.: Move Player an. seq.: 3.4.2, 3.4.4 sk. seq.: 5.3.3 sk. com.: 5.4.3

Use-case neve	Move to Cistern
Rövid leírás	Egy játékost egy ciszternára léptet.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy játékost megkísérel egy csőről egy szomszédos ciszternára mozgatni.</p> <p>2. A játékost eltávolítja az adott csőről.</p> <p>3. Ráteszi a játékost a ciszternára.</p>
Kapcsolatok:	an. req.: D25, D30 an. u-c.: Move Player an. seq.: 3.4.2, 3.4.4 sk. seq.: 5.3.4 sk. com.: 5.4.4

Use-case neve	Move to Source
Rövid leírás	Egy játékost egy forrásra léptet.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy játékost megkísérel egy csőről egy szomszédos forrásra mozgatni.</p> <p>2. A játékost eltávolítja az adott csőről.</p> <p>3. Ráteszi a játékost a forrásra.</p>
Kapcsolatok:	an. req.: D25, D30 an. u-c.: Move Player an. seq.: 3.4.2, 3.4.4 sk. seq.: 5.3.5 sk. com.: 5.4.5

Use-case neve	Fix broken Pump
Rövid leírás	Megjavítatja egy szerelővel a pumpát, amin éppen áll.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy szerelővel megkísérli megjavítatni a törött pumpát, amin áll.</p> <p>2. A szerelő sikeresen megjavítja.</p> <p>3. Véget ér a szerelő köre (turn).</p>
Kapcsolatok:	an. req.: D08, D19, D30 an. u-c.: Repair Part an. seq.: 3.4.5, 3.4.8 sk. seq.: 5.3.6 sk. com.: 5.4.6

Use-case neve	Change directions of not broken Pump
Rövid leírás	Megváltoztatja egy működő pumpának a bemenetét és kimenetét egy játékossal.
Aktorok	Test Controller, Tester
Forgatókönyv	<p>1. Egy szerelővel megkísérli átállítatni a működő pumpát, amin áll.</p> <p>2. Bekéri az új be és kimenetet a Testertől.</p> <p>3. A szerelő sikeresen átállítja a pumpát.</p> <p>4. Véget ér a szerelő köre (turn).</p> <p>5. Egy szabotőrrel megkísérli átállítatni a működő pumpát, amin áll.</p> <p>6. Bekéri az új be és kimenetet a Testertől.</p> <p>7. A szabotőr sikeresen átállítja a pumpát.</p> <p>8. Véget ér a szabotőr köre (turn).</p>
Kapcsolatok:	<p>an. req.: D04, D08, D12, D19, D30</p> <p>an. u-c.: Change Pump direction</p> <p>an. seq.: 3.4.5, 3.4.7, 3.4.8</p> <p>sk. seq.: 5.3.7</p> <p>sk. com.: 5.4.7</p>

Use-case neve	Change directions of broken Pump
Rövid leírás	Megváltoztatja egy nem működő pumpának a bemenetét és kimenetét.
Aktorok	Test Controller, Tester
Forgatókönyv	<p>1. Egy szabotőrrel megkísérli átállítatni a nem működő pumpát, amin áll.</p> <p>2. Bekéri az új be és kimenetet a Testertől.</p> <p>3. A szabotőr sikeresen átállítja a pumpát.</p> <p>4. Véget ér a szabotőr köre (turn).</p> <p>5. Egy szerelővel megkísérli átállítatni a nem működő pumpát, amin áll.</p> <p>6. A szerelő nem állítja át, hanem sikeresen megjavítja.</p> <p>7. Véget ér a szerelő köre (turn).</p>
Kapcsolatok:	<p>an. req.: D04, D08, D12, D19, D30</p> <p>an. u-c.: Change Pump direction</p> <p>an. seq.: 3.4.5, 3.4.7, 3.4.8</p> <p>sk. seq.: 5.3.8</p> <p>sk. com.: 5.4.8</p>

Use-case neve	Cross Cisterns
Rövid leírás	Egy játékoszt a ciszternáról, amin áll, egy másikra helyezi át.
Aktorok	Test Controller
Forgatókönyv	<ol style="list-style-type: none"> 1. A játékoszt leveszi a jelenlegi ciszternáról, amin áll. 2. A játékoszt felteszi a következő ciszternára.
Kapcsolatok:	<p>an. req.: D30, D32 (saját opcionális követelményünk)</p> <p>an. u-c.: Cross Cisterns</p> <p>an. seq.: 3.4.5, 3.4.6</p> <p>sk. seq.: 5.3.9</p> <p>sk. com.: 5.4.9</p>

Use-case neve	Break not broken Pipe
Rövid leírás	Kilyukasztatja egy szabotőrrel a nem lyukas csövet, amin éppen áll.
Aktorok	Test Controller
Forgatókönyv	<ol style="list-style-type: none"> 1. Egy szabotőrrel megkísérel kilyukasztatni egy nem lyukas csövet, amin áll. 2. A szabotőr sikeresen kilyukasztja a csövet. 3. Véget ér a szabotőr köre (turn).
Kapcsolatok:	<p>an. req.: D12, D19, D30</p> <p>an. u-c.: Hole Pipe</p> <p>an. seq.: 3.4.5, 3.4.9</p> <p>sk. seq.: 5.3.10</p> <p>sk. com.: 5.4.10</p>

Use-case neve	Break broken Pipe
Rövid leírás	Megkísérli kilyukasztatni egy szabotőrrel a lyukas csövet, amin éppen áll.
Aktorok	Test Controller
Forgatókönyv	<ol style="list-style-type: none"> 1. Egy szabotőrrel megkísérel kilyukasztatni egy már lyukas csövet, amin áll. 2. A cső lyukas marad.
Kapcsolatok:	<p>an. req.: D12, D19, D30</p> <p>an. u-c.: Hole Pipe</p> <p>an. seq.: 3.4.5, 3.4.9</p> <p>sk. seq.: 5.3.11</p> <p>sk. com.: 5.4.11</p>

Use-case neve	Fix not broken Pipe
Rövid leírás	Megkíséri megjavítatni egy szerelővel a nem lyukas csövet, amin éppen áll.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy szerelővel megkísérel megjavítatni egy nem lyukas csövet, amin áll.</p> <p>2. A cső épp marad.</p>
Kapcsolatok:	an. req.: D08, D19, D30 an. u-c.: Repair Part an. seq.: 3.4.5, 3.4.10 sk. seq.: 5.3.12 sk. com.: 5.4.12

Use-case neve	Fix broken Pipe
Rövid leírás	Egy szerelővel megjavítatja a lyukas csövet, amin éppen áll..
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy szerelővel megkísérel megjavítatni egy lyukas csövet, amin áll.</p> <p>2. A szerelő sikeresen megjavítja a csövet.</p> <p>3. Véget ér a szerelő köre (turn).</p>
Kapcsolatok:	an. req.: D08, D19, D30 an. u-c.: Repair Part an. seq.: 3.4.5, 3.4.10 sk. seq.: 5.3.13 sk. com.: 5.4.13

Use-case neve	Relocate relocatable Pipe
Rövid leírás	Megkísérlel egy mozgatható csövet felvenni, és áthelyezni egy szerelő tárolójába.
Aktorok	Test Controller, Tester
Forgatókönyv	<p>1. Egy olyan csövet próbál felvenni, amiben nem folyik víz, nem törött és játékos sem áll rajta.</p> <p>1.A.1. A szerelő tárolója üres.</p> <p>1.A.2. A cső bekerül a szerelő tárolójába (előző helyéről pedig eltűnik).</p> <p>1.B.1. A cső helyben marad (nem fér a szerelő tárolójába, mert már van nála egy part).</p>
Kapcsolatok:	an. req.: D05, D20, D24, D30 an. u-c.: Store Part an. seq.: 3.4.12 sk. seq.: 5.3.14 sk. com.: 5.4.14

Use-case neve	Relocate unrelocatable Element
Rövid leírás	Megkísérel egy nem mozgatható elemet felvenni.
Aktorok	Test Controller, Tester
Forgatókönyv	<p>1. Egy nem felvehető elemet próbál felvenni (nem cső, folyik benne víz, törött vagy áll rajta játékos).</p> <p>2. Az elem helyben marad.</p>
Kapcsolatok:	an. req.: D05, D20, D24, D30 an. u-c.: Store Part an. seq.: 3.4.12 sk. seq.: 5.3.15 sk. com.: 5.4.15

Use-case neve	Pick up Pump when possible
Rövid leírás	Pumpát helyez egy ciszternán álló szerelő tárolójába, ha az üres.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy ciszternán álló, üres tárolóval rendelkező szerelőnek kísérel meg pumpát adni.</p> <p>2. A szerelő tárolójához egy új pumpát ad.</p>
Kapcsolatok:	an. req.: D10, D24, D30 an. u-c.: Produce Part, Store Part an. seq.: 3.4.13 sk. seq.: 5.3.16 sk. com.: 5.4.16

Use-case neve	Pick up Pump when impossible
Rövid leírás	Pumpát kísérel meg helyezni egy ciszternán álló szerelő tárolójába, amikor az nem üres.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy ciszternán álló, nem üres tárolóval rendelkező szerelőnek kísérel meg pumpát adni.</p> <p>2. A szerelő nem kap új pumpát, és a tárolója megőrzi a korábban felvett part-ot.</p>
Kapcsolatok:	an. req.: D10, D24, D30 an. u-c.: Produce Part, Store Part an. seq.: 3.4.13 sk. seq.: 5.3.17 sk. com.: 5.4.17

Use-case neve	Place Pipe when placeable
Rövid leírás	Egy szerelő tárolójából kiveszi a tárolt csövet, és leteszi, amikor ez lehetséges.
Aktorok	Test Controller
Forgatókönyv	<p>1. Megkísérel egy csővel rendelkező szerelő tárolójából kivenni a csövet, és lehelyezni azt egy alkalmas üres szomszédos helyre (legfeljebb 2 szomszédja lesz).</p> <p>2. A cső kikerül a szerelő tárolójából. (A tárolója üres lesz.)</p> <p>3. A cső a választott helyre kerül.</p> <p>4. Véget ér a szerelő köre (turn).</p>
Kapcsolatok:	an. req.: D05, D21, D23, D30 an. u-c.: Relocate Pipe an. seq.: 3.4.14, 3.4.15 sk. seq.: 5.3.18 sk. com.: 5.4.18

Use-case neve	Place Pipe when not placeable
Rövid leírás	Egy szerelő tárolójából egy csövet kísérel meg lerakni, amikor ez nem lehetséges.
Aktorok	Test Controller
Forgatókönyv	<p>1. Megkísérel lerakni egy csövet egy szerelő tárolójából, amikor ez nem lehetséges (nincs nála cső, a választott szomszédos hely nem üres vagy több mint 2 szomszédja lenne a lerakott csőnek).</p> <p>2. A cső a szerelő tárolójában marad.</p>
Kapcsolatok:	an. req.: D05, D21, D23, D30 an. u-c.: Relocate Pipe an. seq.: 3.4.14, 3.4.15 sk. seq.: 5.3.19 sk. com.: 5.4.19

Use-case neve	Place Pump when placeable
Rövid leírás	Egy szerelő tárolójából kiveszi a tárolt pumpát, és leteszi, amikor ez lehetséges.
Aktorok	Test Controller
Forgatókönyv	<p>1. Megkísér egy pumpával rendelkező szerelő tárolójából kivenni a pumpát, és lehelyezni azt egy alkalmas cső helyére (a csövön nem áll játékos, és van 2 szomszédos csöve, ami a pumpa be és kimenete lesz).</p> <p>2. A pumpa kikerül a szerelő tárolójából. (A tárolója üres lesz.)</p> <p>3. A pumpa a kiválasztott cső helyére kerül. A csövet felülírja, tehát az meg fog szűnni létezni.</p> <p>4. Véget ér a szerelő köre (turn).</p>
Kapcsolatok:	an. req.: D11, D22, D23, D30 an. u-c.: Place Pump an. seq.: 3.4.14, 3.4.16 sk. seq.: 5.3.20 sk. com.: 5.4.20

Use-case neve	Place Pump when not placeable
Rövid leírás	Egy szerelő tárolójából egy pumpát kísérel meg lerakni, amikor ez nem lehetséges.
Aktorok	Test Controller
Forgatókönyv	<p>1. Megkísérlerakni egy pumpát egy szerelő tárolójából, amikor ez nem lehetséges (nincs nála pumpa vagy a célpont elem nem olyan játékos nélküli cső, aminek van 2 szomszédos csöve).</p> <p>2. A pumpa a szerelő tárolójában marad.</p>
Kapcsolatok:	an. req.: D11, D22, D23, D30 an. u-c.: Place Pump an. seq.: 3.4.14, 3.4.16 sk. seq.: 5.3.21 sk. com.: 5.4.21

Use-case neve	Flood not broken Pipe
Rövid leírás	Vizet áraszt egy nem lyukas csőbe.
Aktorok	Test Controller, Tester
Forgatókönyv	<p>1. Vizet enged egy nem lyukas csőbe.</p> <p>2. Víz kerül a csőbe.</p> <p>2.A.1. A csőnek van kimenete.</p> <p>2.A.2. A cső a kimenetére tovább engedi a vizet.</p> <p>2.A.3. Ha a kimenetenek már nincs kimenete, a szabotőrök pontot kapnak. (Sivatagba folyik a víz.)</p> <p>2.B.1. A szabotőrök pontot kapnak. (Sivatagba folyik a víz.)</p>
Kapcsolatok:	an. req.: D07, D09 an. u-c.: Control Waterflow an. seq.: 3.4.20 sk. seq.: 5.3.22 sk. com.: 5.4.22

Use-case neve	Flood broken Pipe
Rövid leírás	Vizet áraszt egy lyukas csőbe.
Aktorok	Test Controller
Forgatókönyv	<p>1. Vizet enged egy lyukas csőbe.</p> <p>2. Víz kerül a csőbe, de nem engedi tovább a kimenetére.</p> <p>3. A szabotőrök pontot kapnak.</p>
Kapcsolatok:	an. req.: D07, D09 an. u-c.: Control Waterflow an. seq.: 3.4.20 sk. seq.: 5.3.23 sk. com.: 5.4.23

Use-case neve	Flood not broken Pump
Rövid leírás	Vizet áraszt egy működő pumpába.
Aktorok	Test Controller
Forgatókönyv	<p>1. Vizet enged egy működő pumpába.</p> <p>2. Víz kerül a pumpába.</p> <p>2.A.1. A pumpának van kimenete.</p> <p>2.A.2. A pumpa a kimenetére tovább engedi a vizet.</p> <p>2.A.3. Ha a kimenetnek már nincs kimenete, a szabotőrök pontot kapnak. (Sivatagba folyik a víz.)</p> <p>2.B.1. A szabotőrök pontot kapnak. (Sivatagba folyik a víz.)</p>
Kapcsolatok:	an. req.: D02, D03, D04 an. u-c.: Control Waterflow an. seq.: 3.4.21 sk. seq.: 5.3.24 sk. com.: 5.4.24

Use-case neve	Flood broken Pump
Rövid leírás	Vizet áraszt egy nem működő pumpába.
Aktorok	Test Controller
Forgatókönyv	<ol style="list-style-type: none"> 1. Vizet enged egy törött pumpába. 2. Víz kerül a pumpába, de nem engedi tovább a kimenetére.
Kapcsolatok:	<p>an. req.: D02, D03, D04</p> <p>an. u-c.: Control Waterflow</p> <p>an. seq.: 3.4.21</p> <p>sk. seq.: 5.3.25</p> <p>sk. com.: 5.4.25</p>

Use-case neve	Flood Source
Rövid leírás	Elindítja a vizet egy forrásból.
Aktorok	Test Controller
Forgatókönyv	<ol style="list-style-type: none"> 1. A forrás vizet enged a kimeneteire. 1.A.1. Ha cső van csatlakoztatva a kimenetre, akkor abba enged vizet. 1.A.2. Ha a kimenetnek már nincs kimenete, a szabotőrök pontot kapnak. (Sivatagba folyik a víz.) 1.B.1. Ha üres a kimenete akkor a sivatagba enged vizet. 1.B.2. A szabotőrök pontot kapnak minden üres kimenetért.
Kapcsolatok:	<p>an. req.: D28, D31</p> <p>an. u-c.: Control Waterflow</p> <p>an. seq.: 3.4.19</p> <p>sk. seq.: 5.3.26</p> <p>sk. com.: 5.4.26</p>

Use-case neve	Flood Cistern
Rövid leírás	Vizet folyat egy ciszternába.
Aktorok	Test Controller
Forgatókönyv	<ol style="list-style-type: none"> 1. Vizet enged egy ciszternába. 2. Víz kerül a ciszternába. 3. A szerelők pontot kapnak.
Kapcsolatok:	<p>an. req.: D28</p> <p>an. u-c.: Control Waterflow</p> <p>an. seq.: 3.4.22</p> <p>sk. seq.: 5.3.27</p> <p>sk. com.: 5.4.27</p>

Use-case neve	Produce Pipe
Rövid leírás	Egy ciszterna mellé egy új csövet próbál helyezni az első üres helyre, ha van ilyen, a következő prioritási sorrendben: észak, nyugat, kelet, dél.
Aktorok	Test Controller
Forgatókönyv	<p>1. Egy adott ciszterna mellé pontosan egy új csövet próbál helyezni, ha van olyan irány, amelyikben nincsen szomszédja.</p> <p>1.A.1. Az északra lévő üres helyen jelenik meg az új cső.</p> <p>1.B.1. A nyugatra lévő üres helyen jelenik meg az új cső.</p> <p>1.C.1. A keletre lévő üres helyen jelenik meg az új cső.</p> <p>1.D.1. A délré lévő üres helyen jelenik meg az új cső.</p> <p>1.E.1. A ciszterna mellett nem jelenik meg új cső, mert minden négy irányban van már.</p>
Kapcsolatok:	an. req.: D06, D18, D29 an. u-c.: Produce Part an. seq.: 3.4.17 sk. seq.: 5.3.28 sk. com.: 5.4.28

Use-case neve	Impair Pump
Rövid leírás	Elront egy pumpát.
Aktorok	Test Controller
Forgatókönyv	1. A játék (Game) elrontja a kiválasztott pumpát.
Kapcsolatok:	an. req.: D03, D31 an. u-c.: Impair Pump an. seq.: 3.4.17 sk. seq.: 5.3.29 sk. com.: 5.4.29

5.2 A szkeleton kezelői felületének terve, dialógusok

A 29 teszteset közül egyszerre egyet választhat a tesztelő. 1-29-ig kell megadni egy számot, más bemenetre nem reagál a program a 0 kivételével, ami hatására terminál a program.

Minden hívott függvény kiírja, a nevét és paramétereit, amikor meghívják (pl. fv(p1, p2), illetve kiírja a visszatérésének tényét, a visszatérés előtt (pl. fv returned).

Ha a függvénynek van visszatérési értéke, azt is hozzáfüzi (pl. fv returned true).

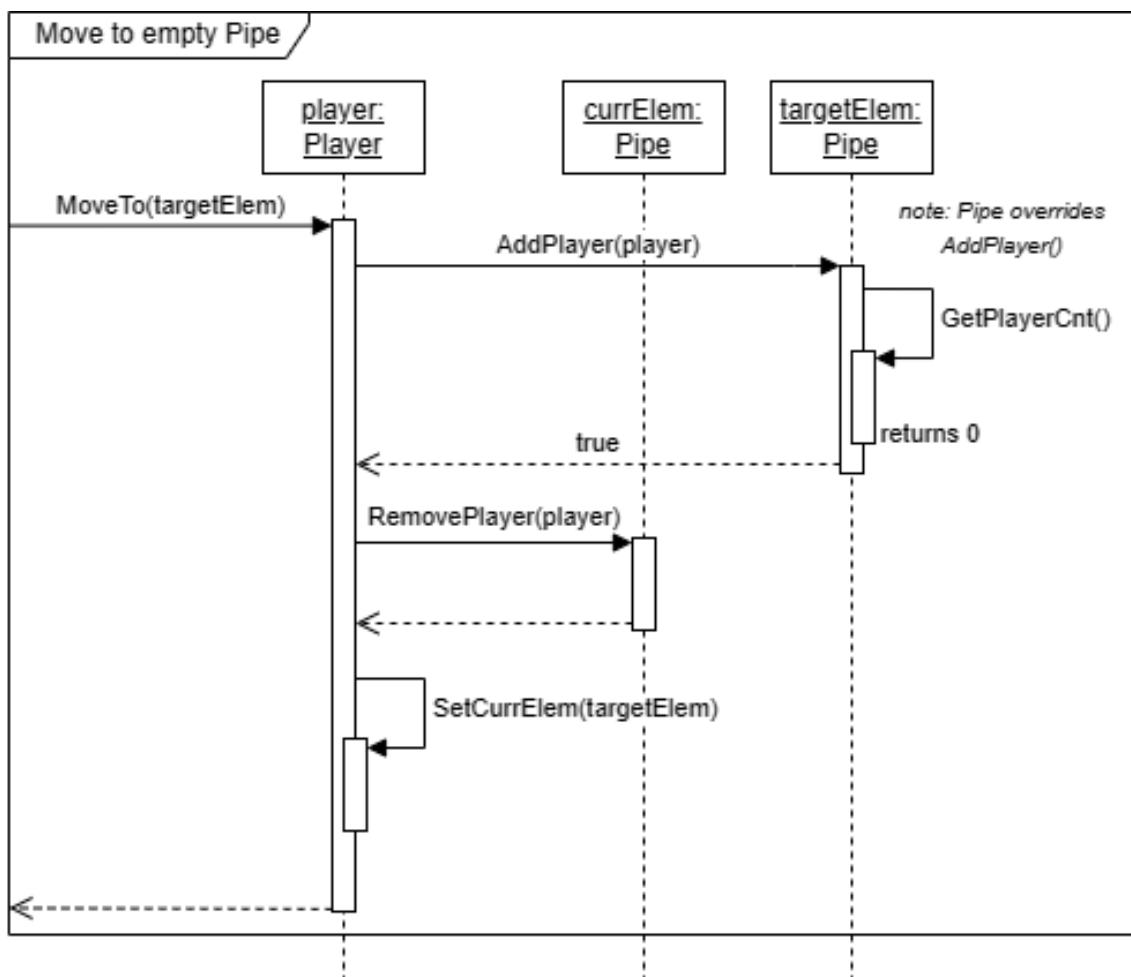
Azon tesztesetek esetén, amikor a tesztelőtől további bemenetre van szükség, addig nem kezdődik el a teszteset futása, amíg ezeket nem adja meg a tesztelő. Ezen bemeneti paraméterek karakterek lesznek, amiket a program jelezni fog, és el fogja magyarázni jelentésüköt.

5.3 Szekvencia diagramok a belső működésre

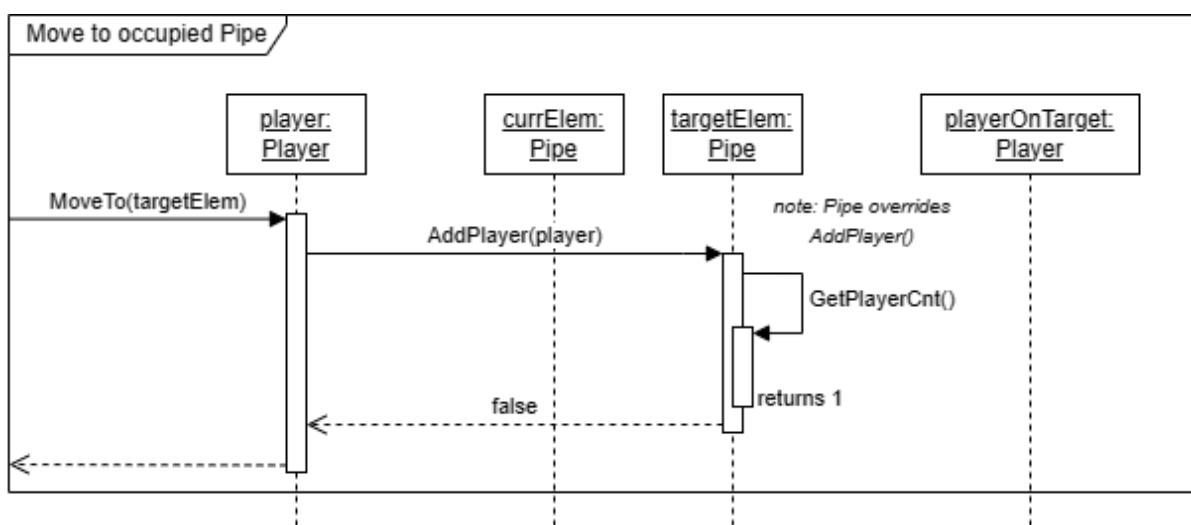
fontos megjegyzések:

- minden esetben SkeletonController indítja a külső hívást
- ha egy use-case-hez több diagram tartozik, akkor azok egymástól függetlenül (külön függvényekben) lesznek megvalósítva,
kivétel, ha „ref” fragmentben hivatkozunk rájuk

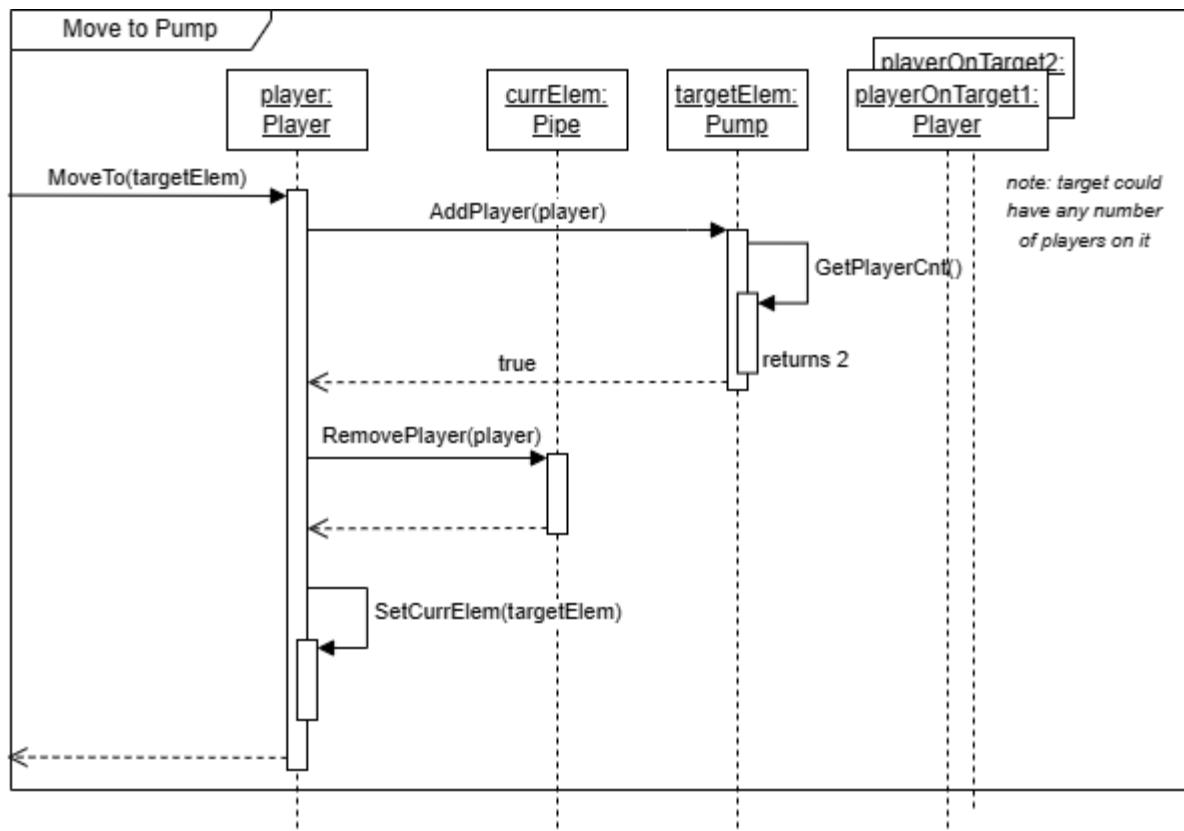
5.3.1 Move to empty Pipe



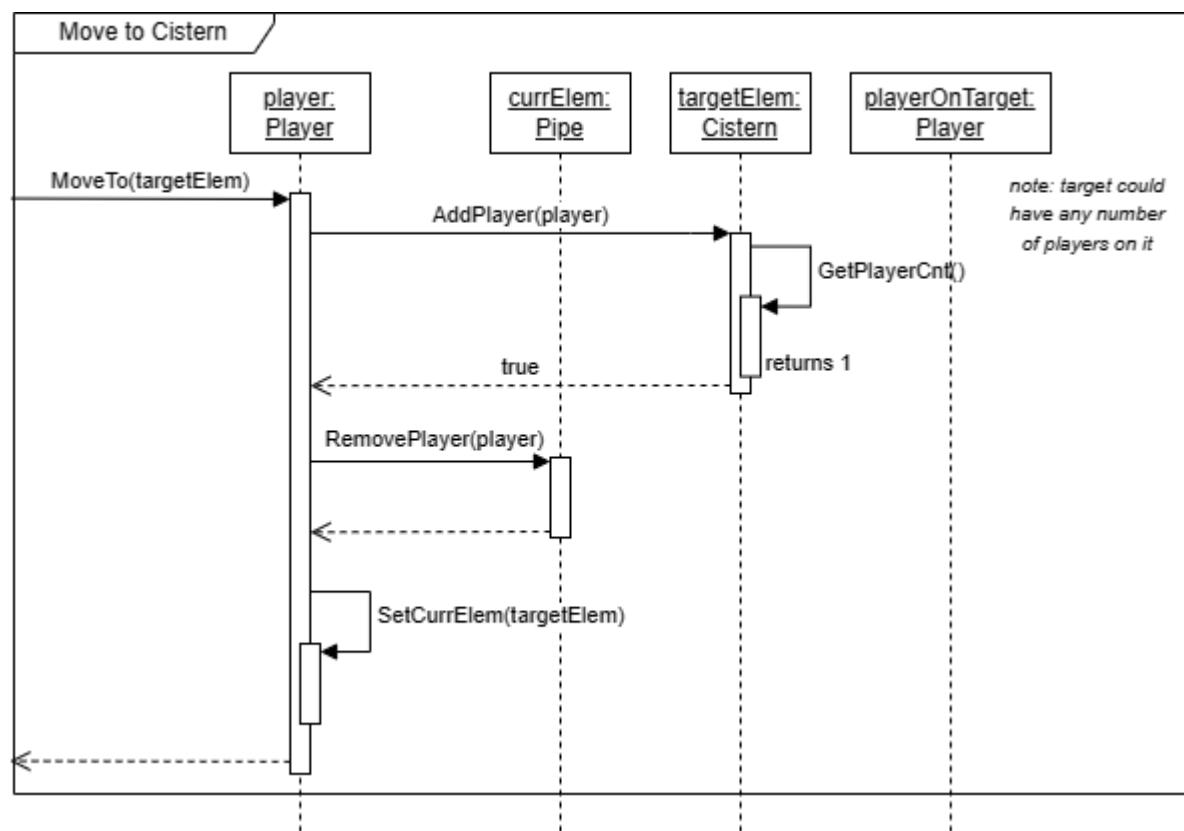
5.3.2 Move to occupied Pipe



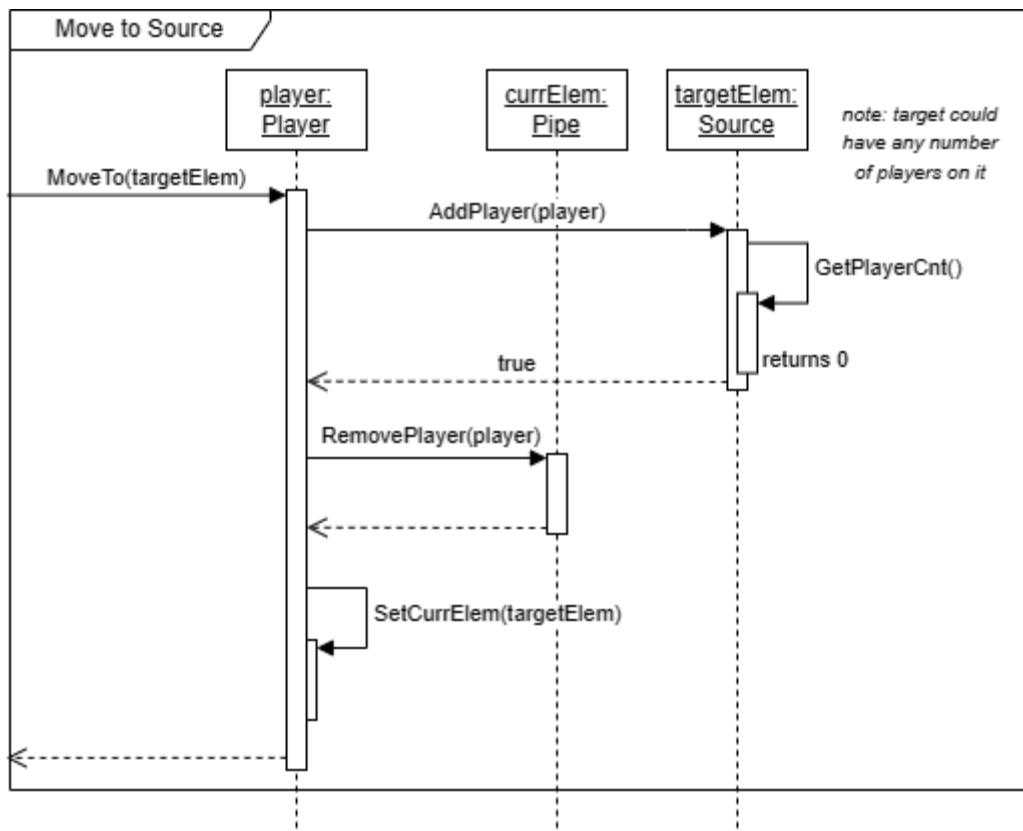
5.3.3 Move to Pump



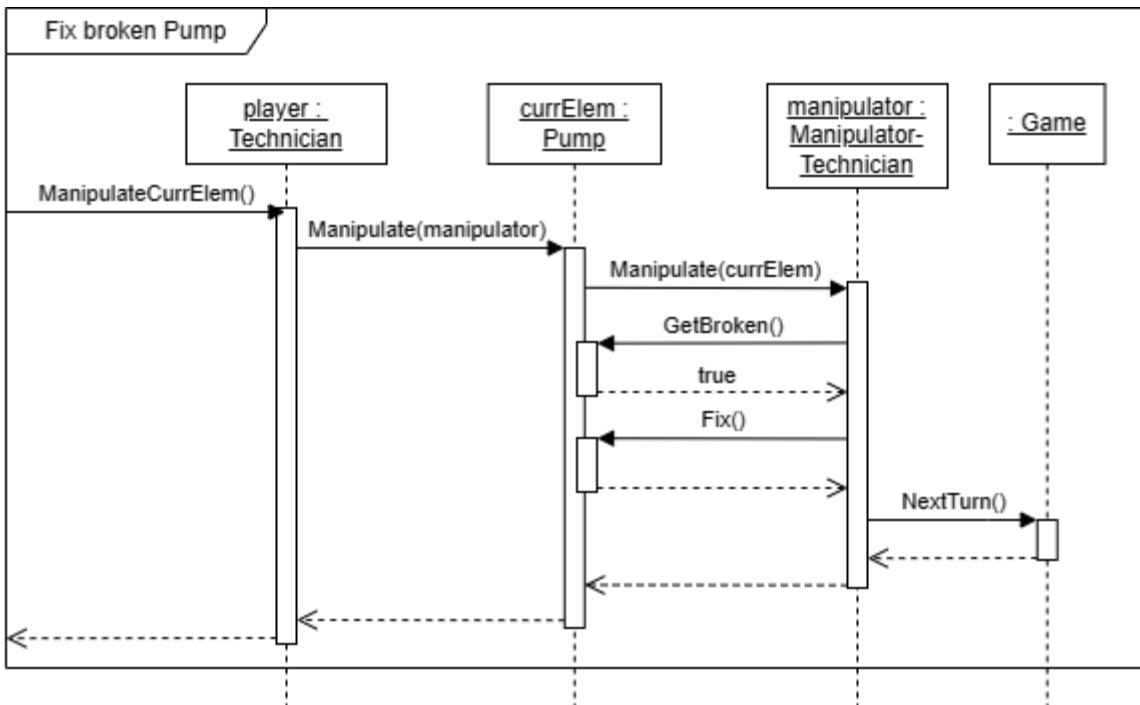
5.3.4 Move to Cistern



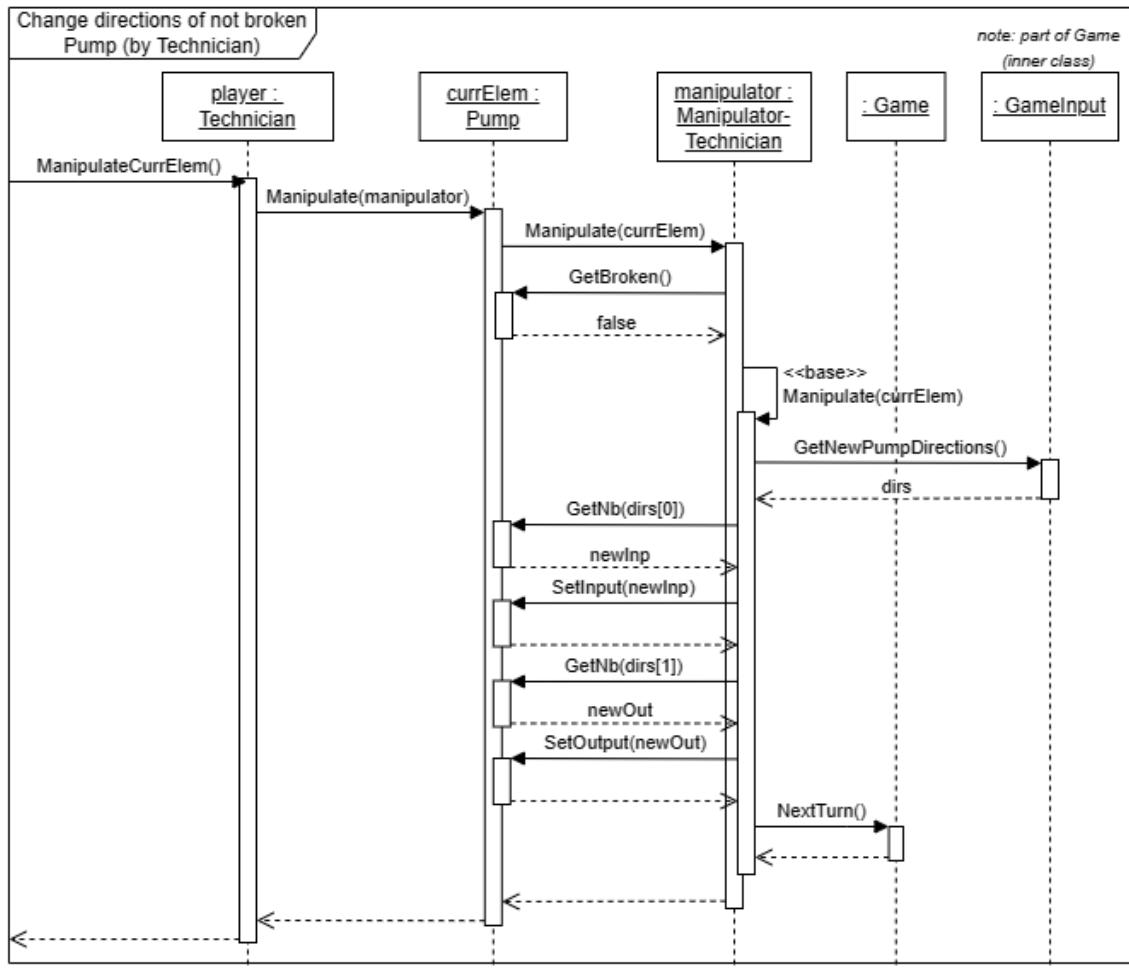
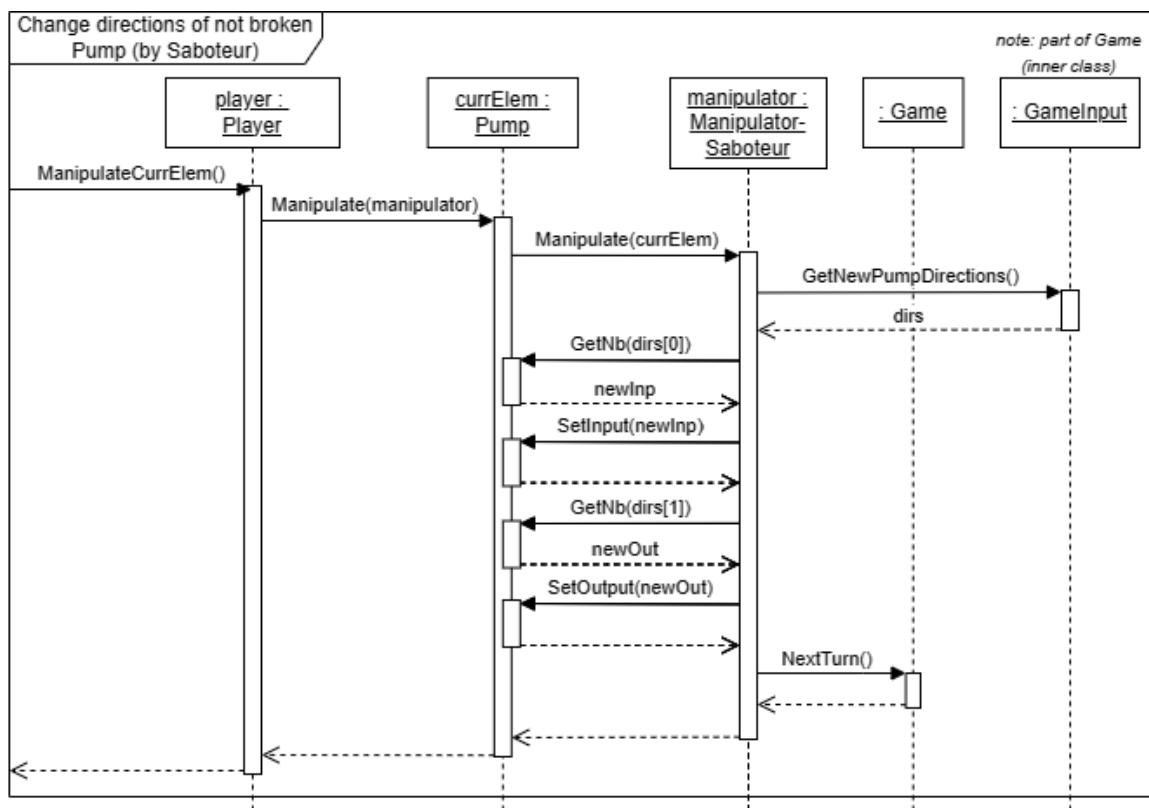
5.3.5 Move to Source



5.3.6 Fix broken Pump



5.3.7 Change directions of not broken Pump



5.3.8 Change directions of broken Pump

Change directions of broken
Pump (by Saboteur)

note: exactly the same as if the Pump wasn't broken

(Saboteur can change the directions regardless if it's broken or not)

ref

Change directions of not broken Pump (by Saboteur)

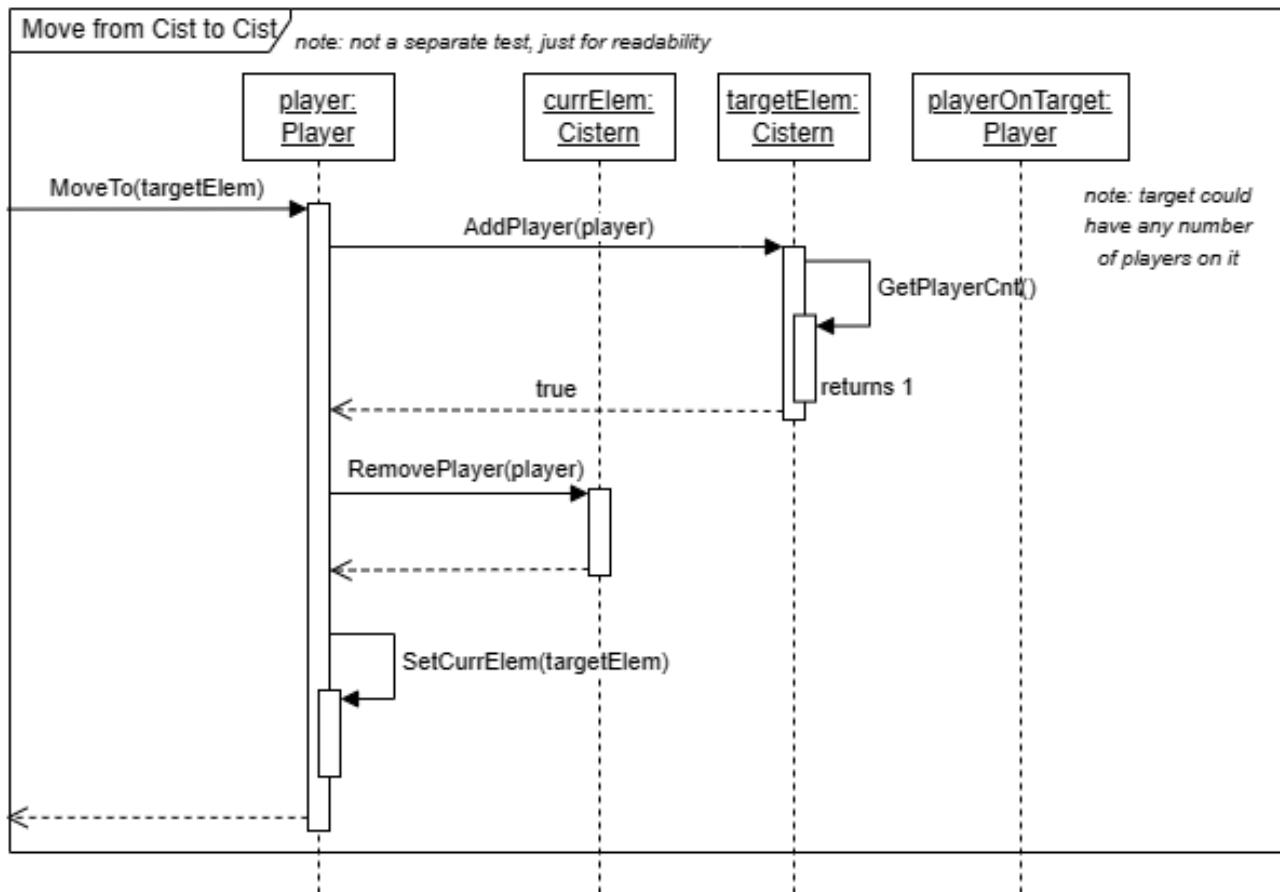
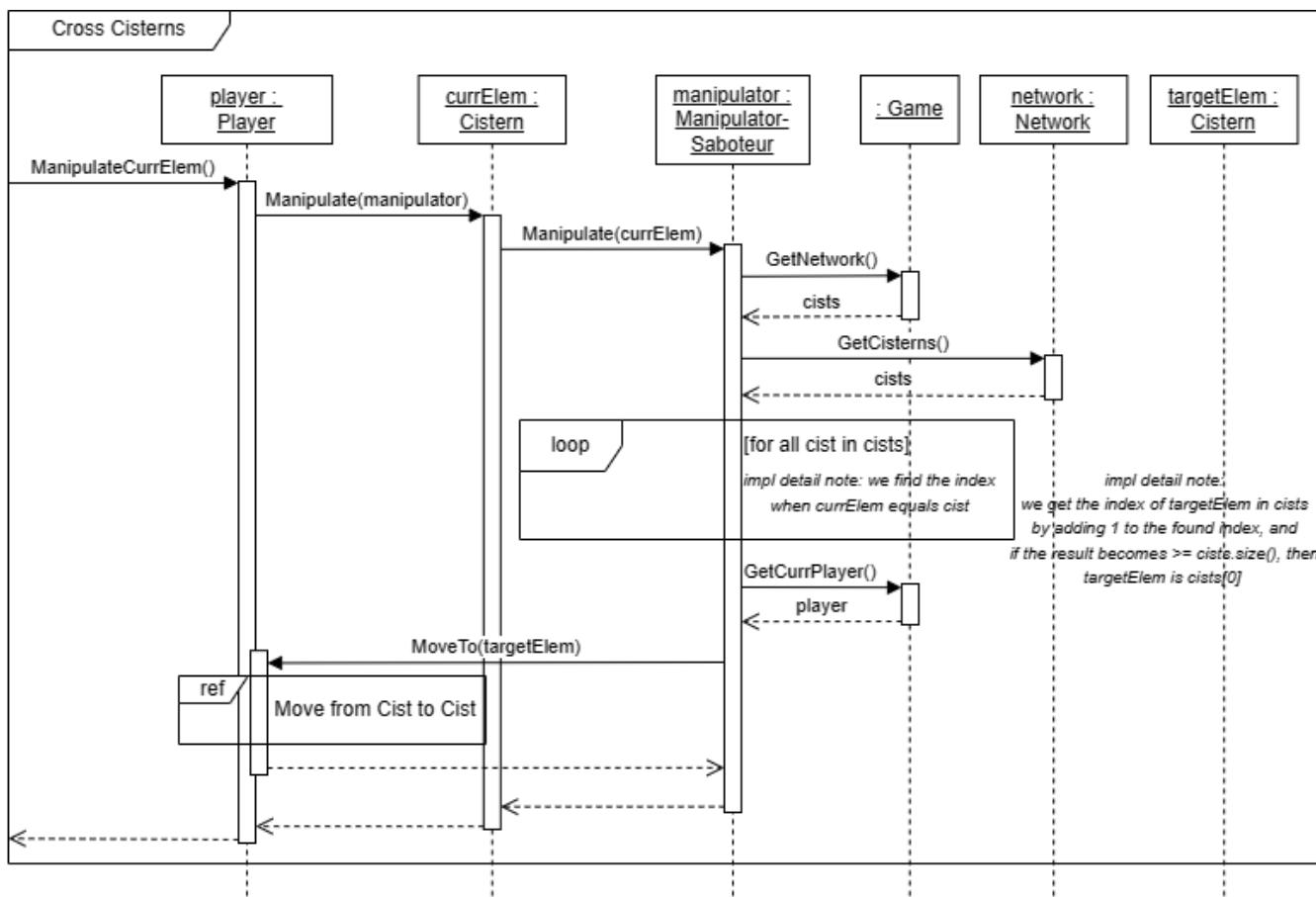
Change directions of broken
Pump (by Technician)

*note: the Technician fixes broken Pumps,
instead of changing their directions*

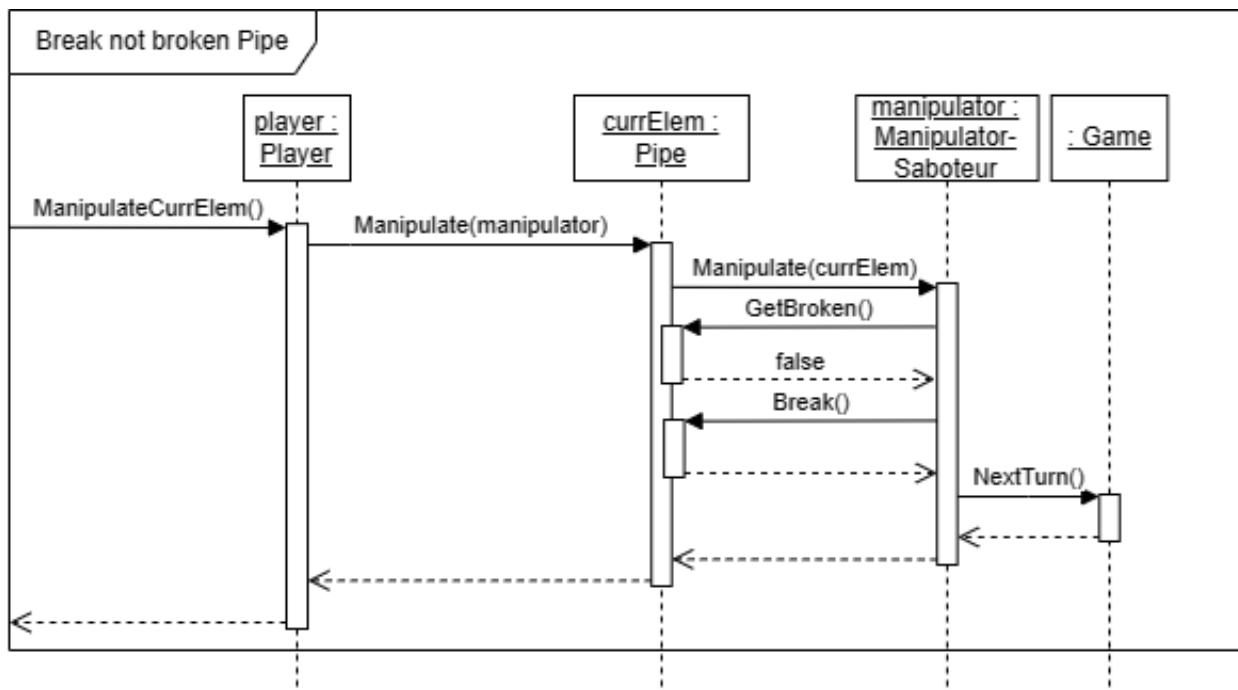
ref

Fix broken Pump

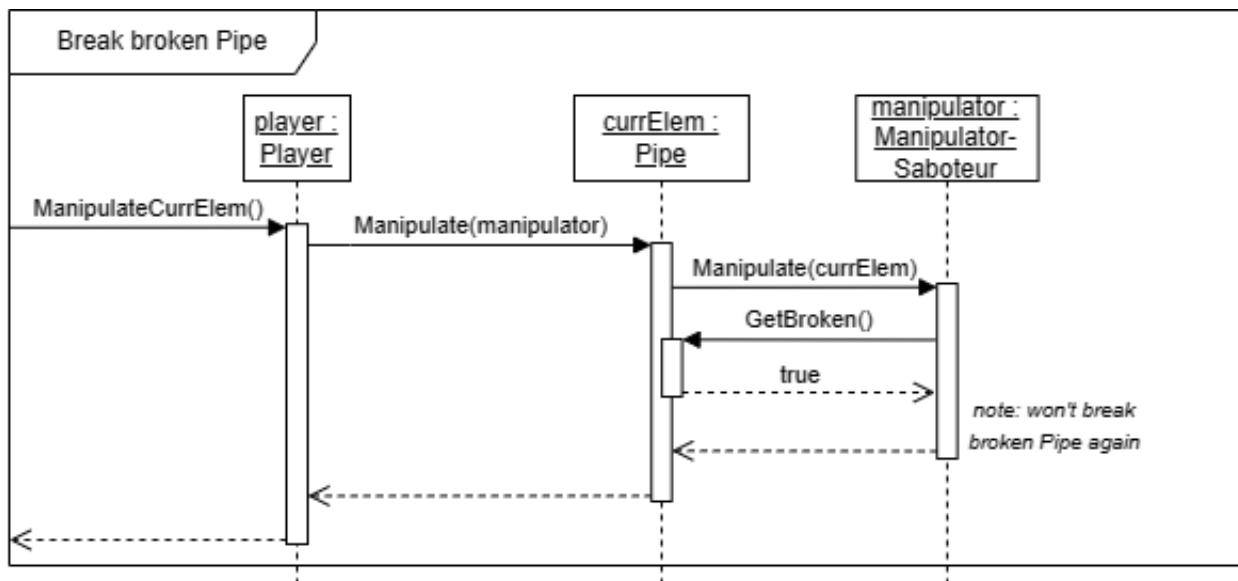
5.3.9 Cross Cisterns



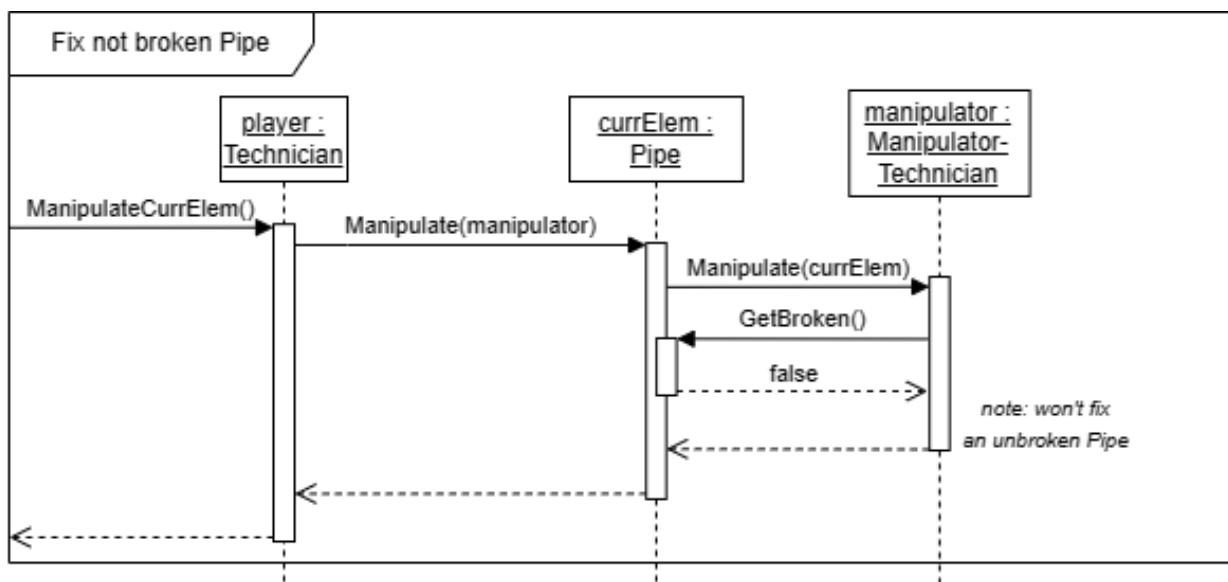
5.3.10 . Break not broken Pipe



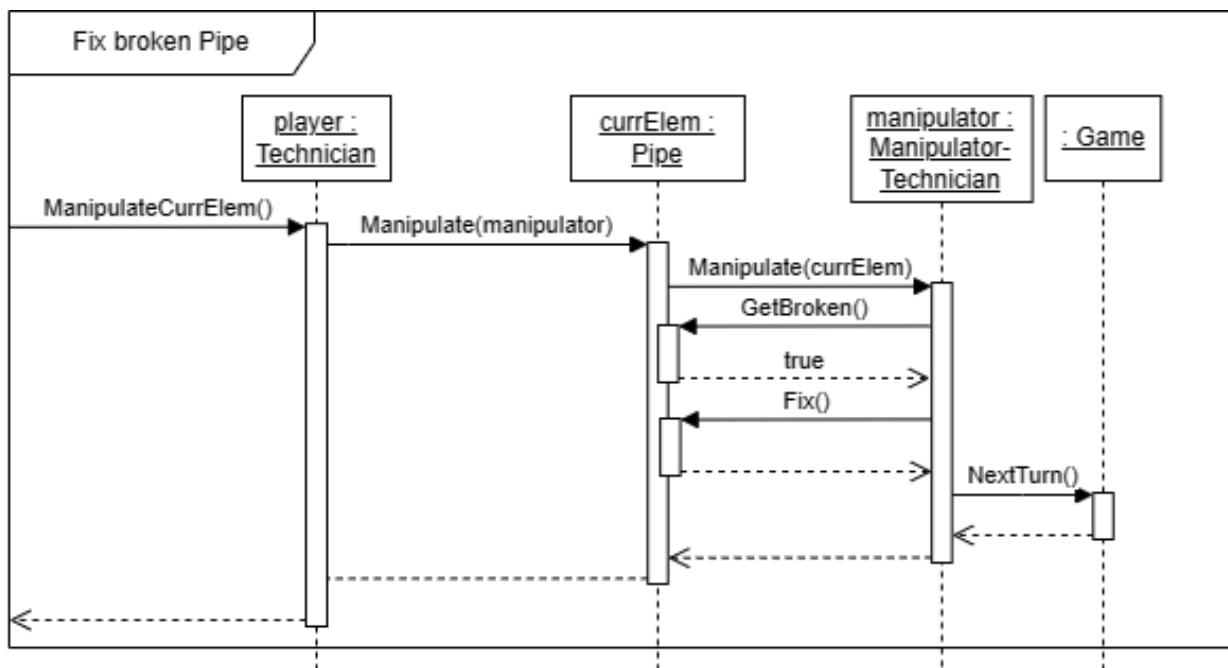
5.3.11 Break broken Pipe



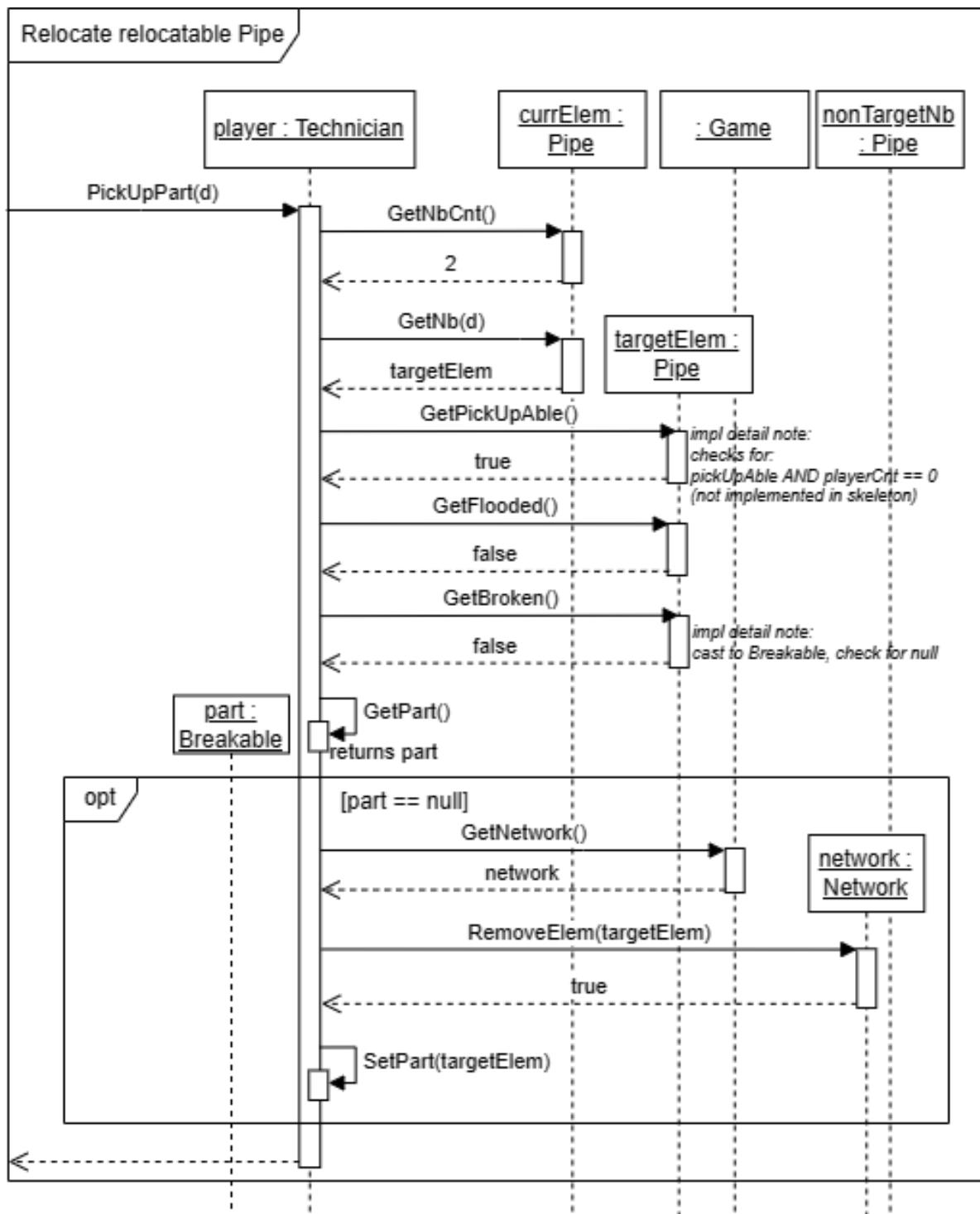
5.3.12 Fix not broken Pipe



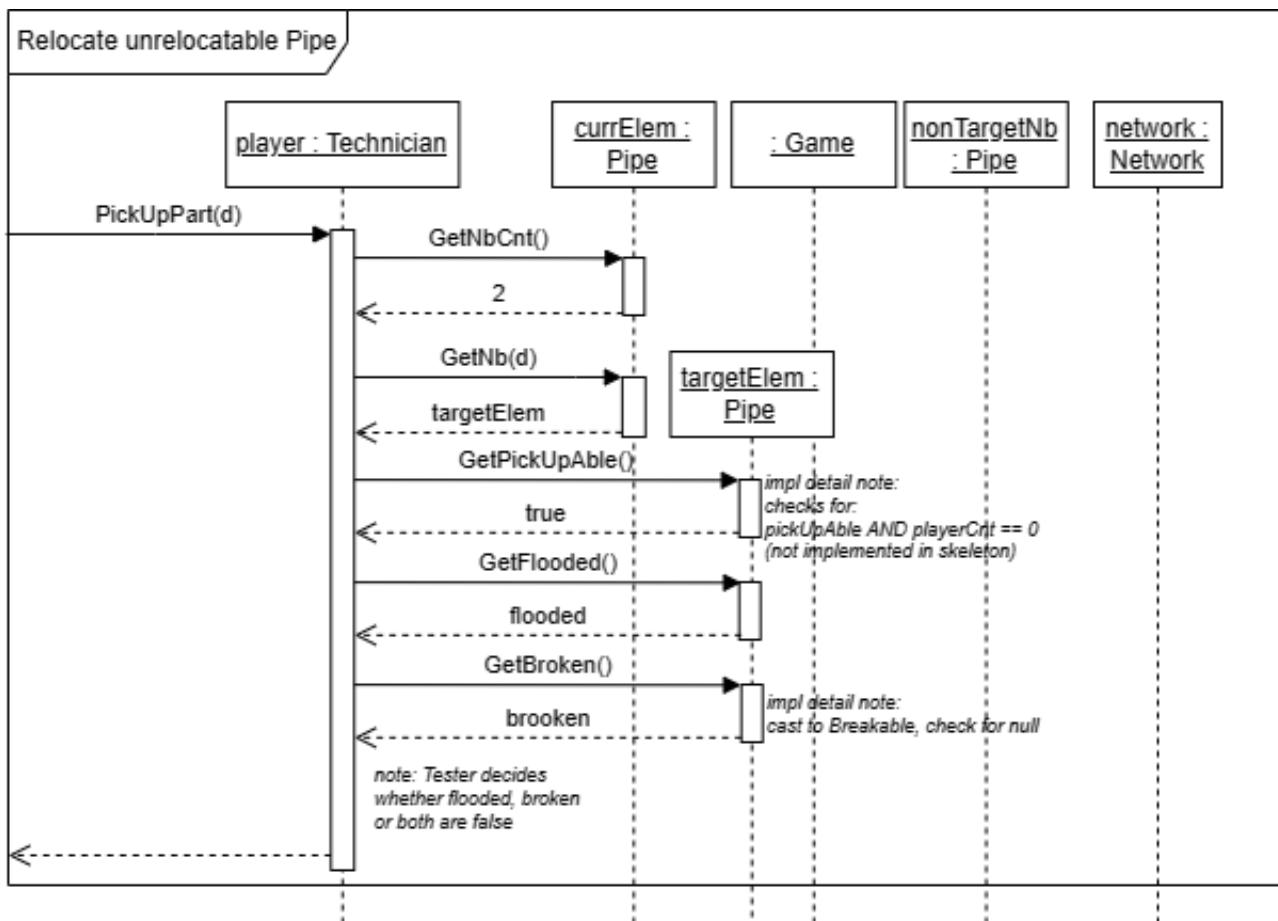
5.3.13 Fix broken Pipe



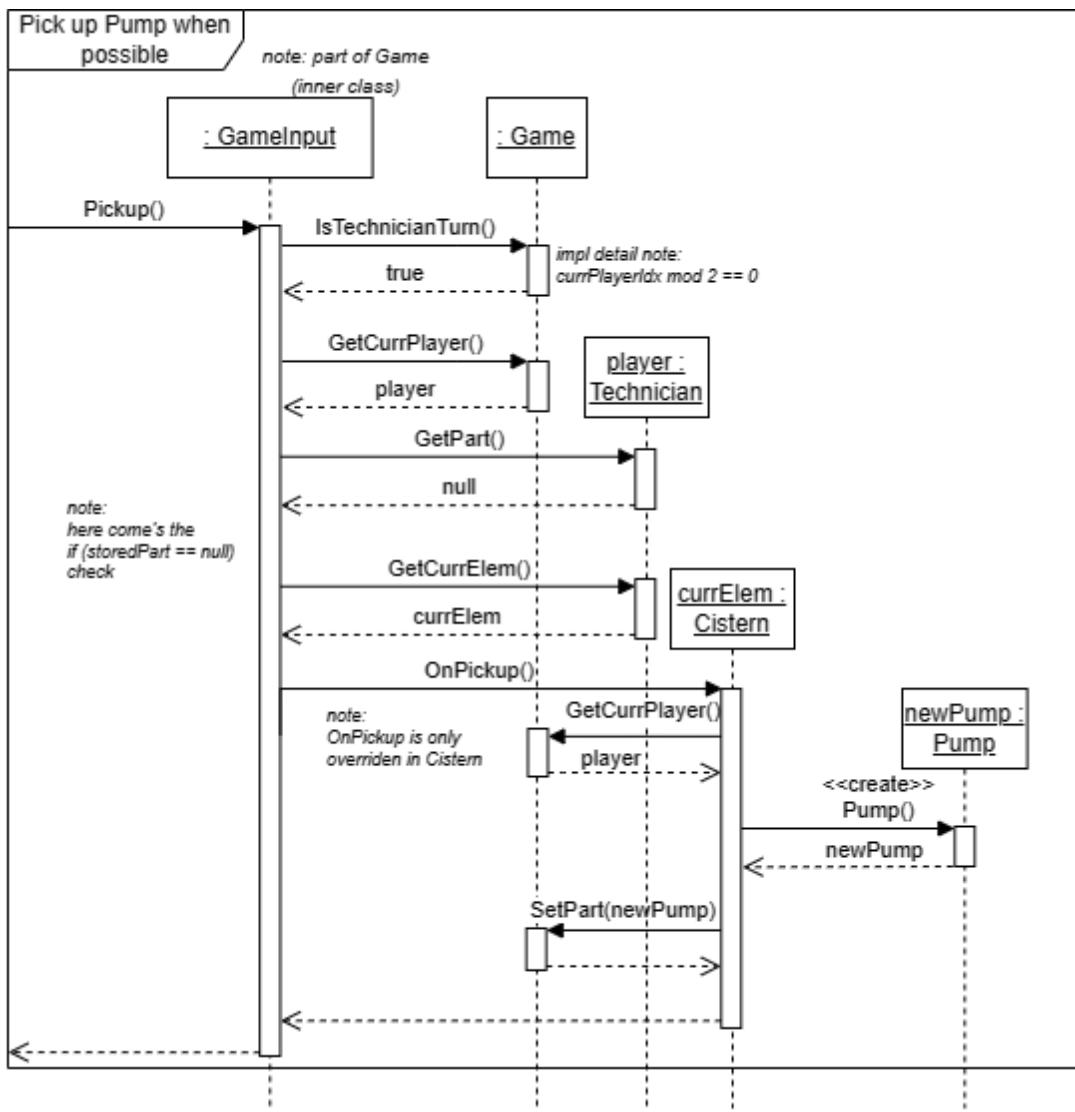
5.3.14 Relocate relocatable Pipe



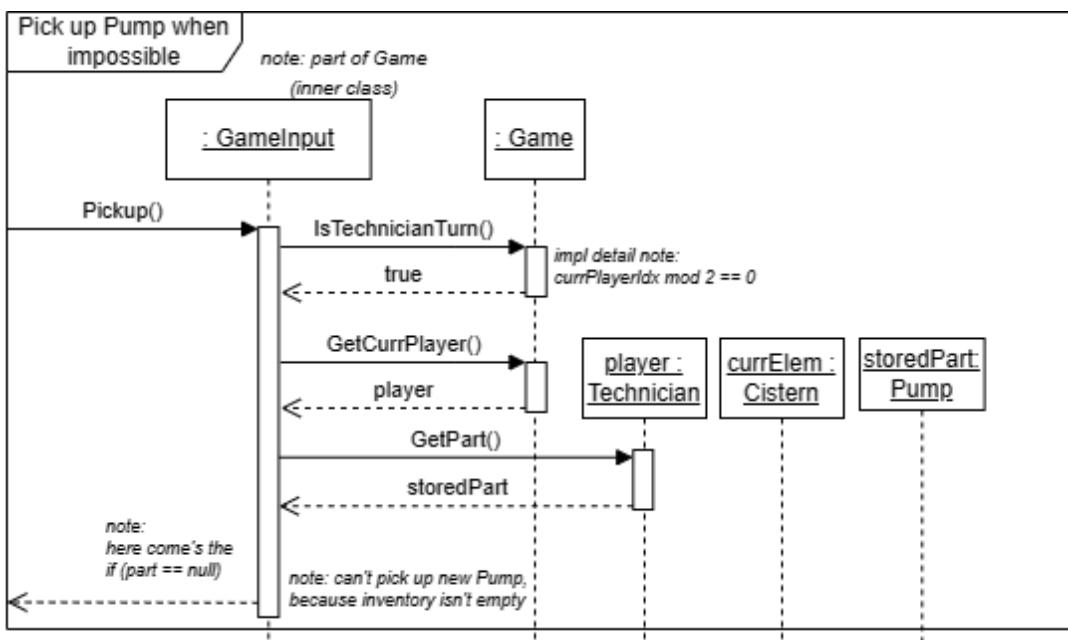
5.3.15 Relocate unrelocatable Element



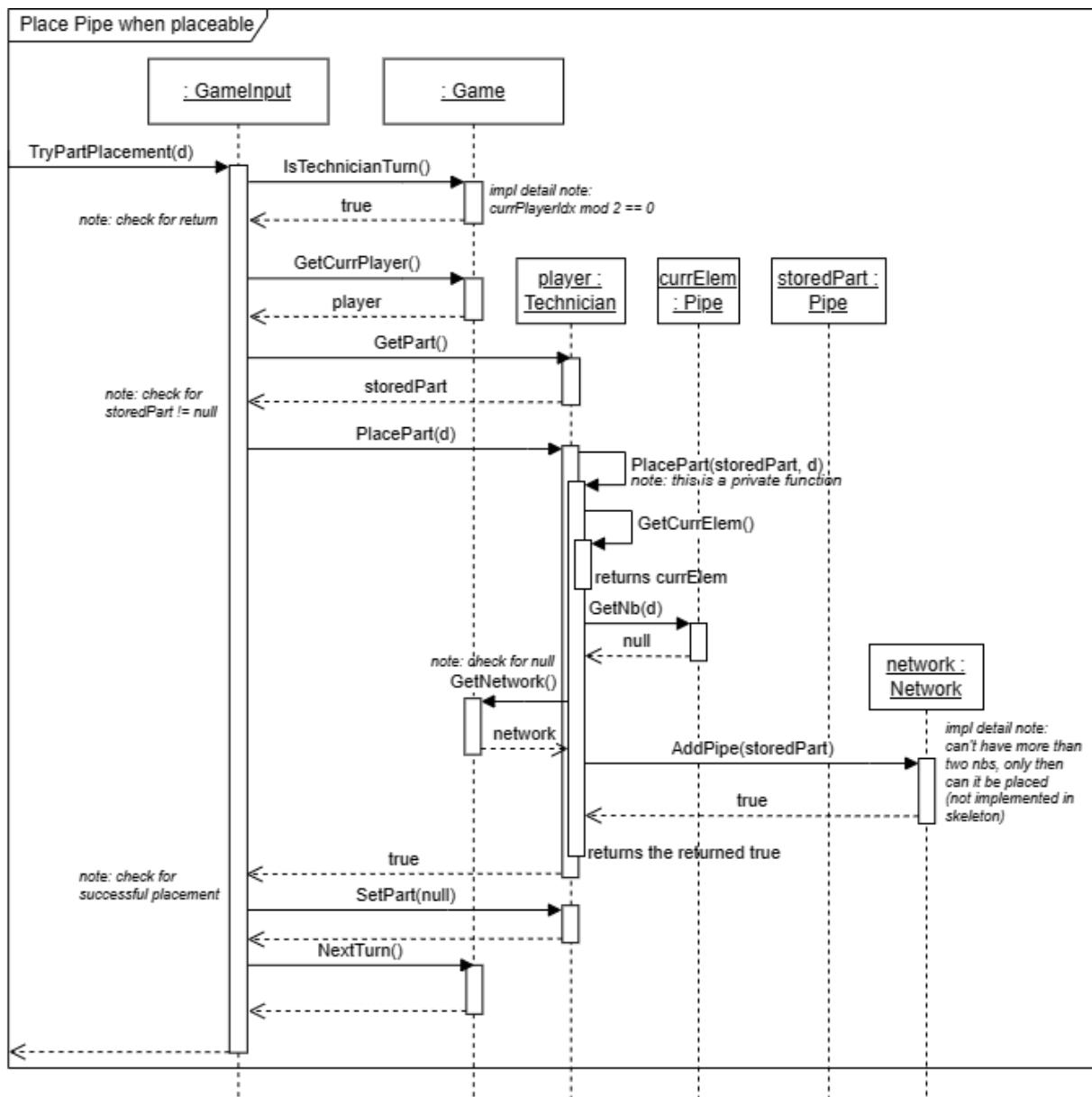
5.3.16 Pick up Pump when possible



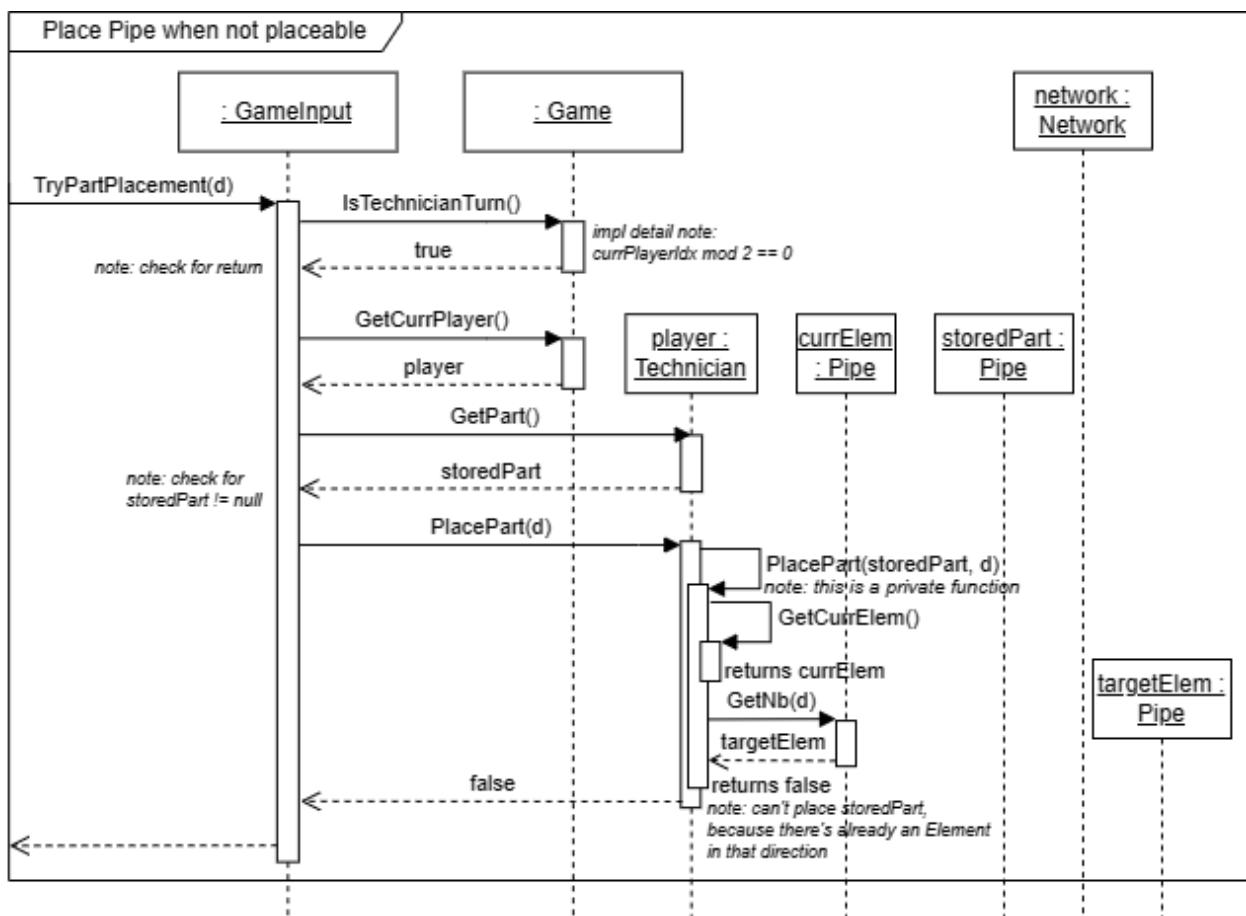
5.3.17 Pick up Pump when impossible



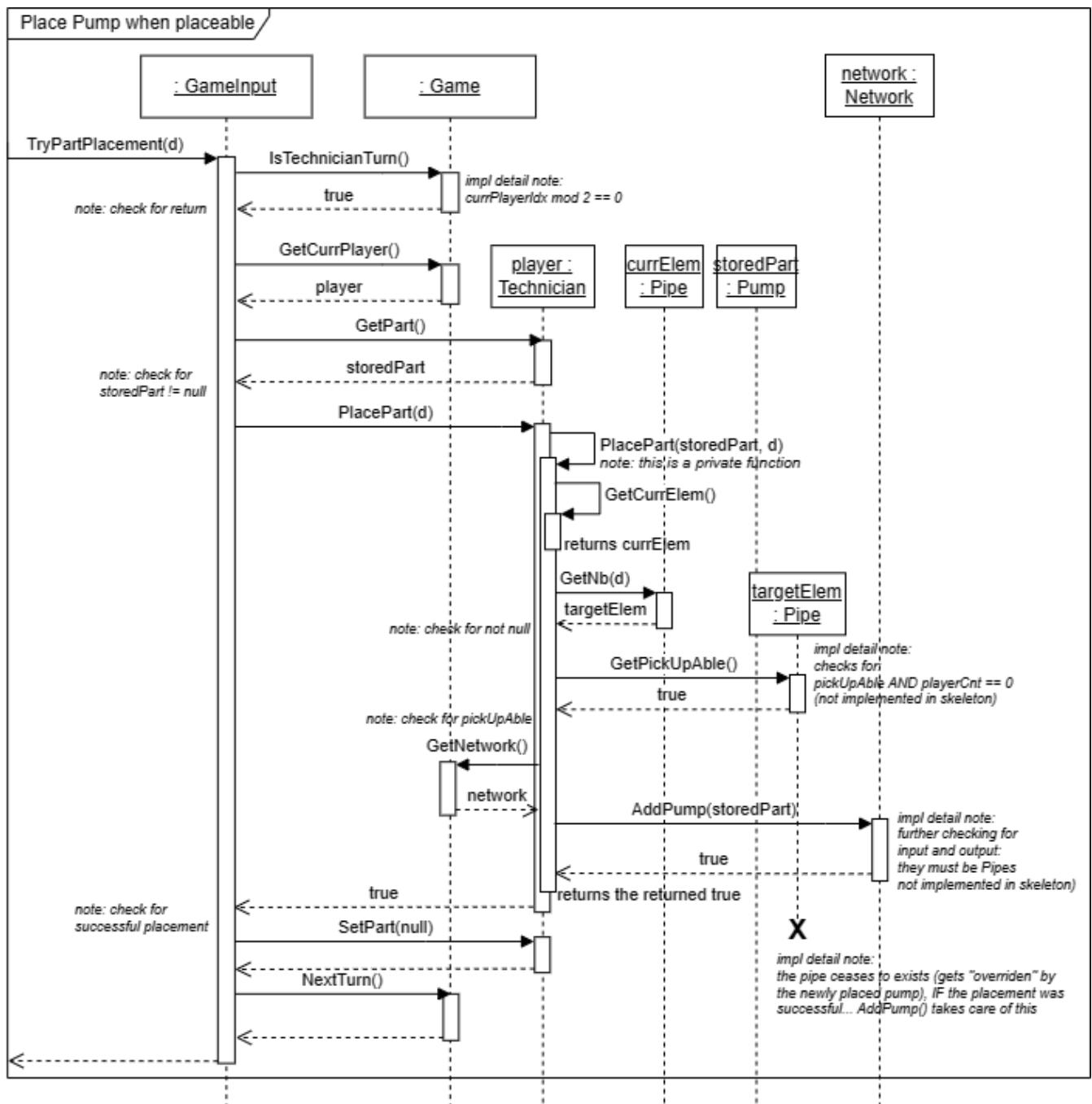
5.3.18 Place Pipe when placeable



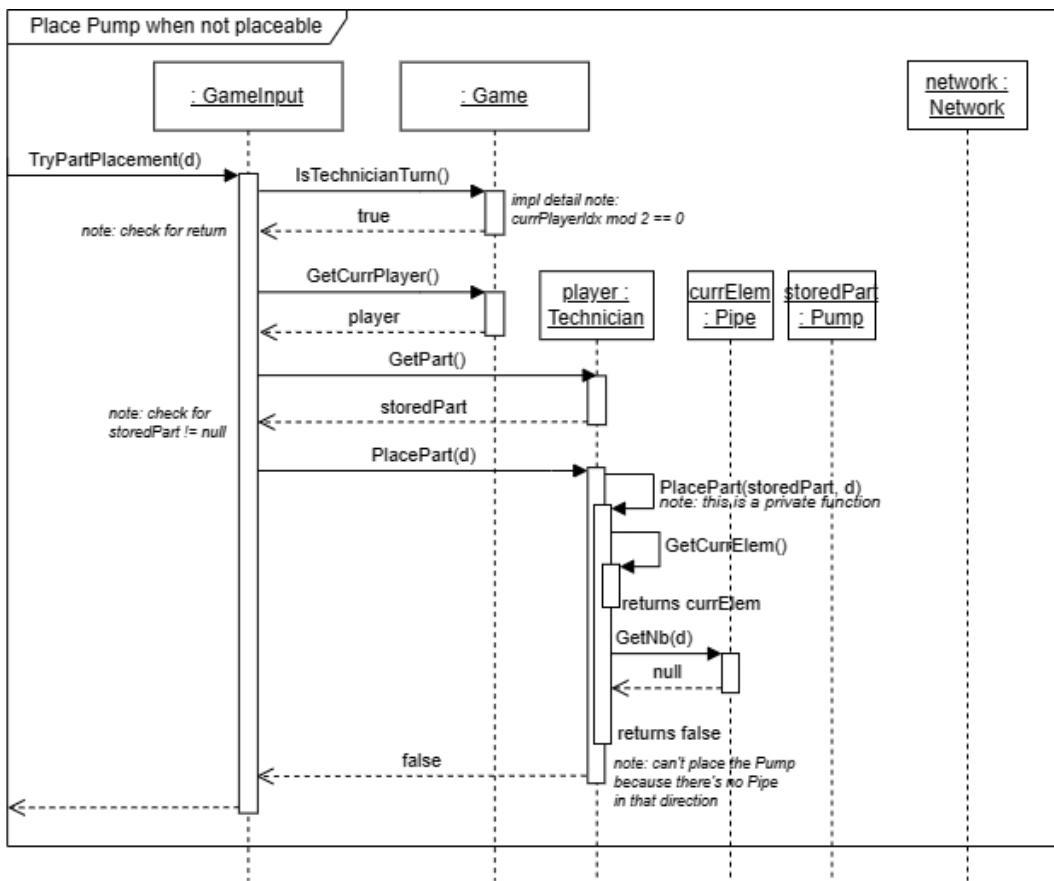
5.3.19 Place Pipe when not placeable



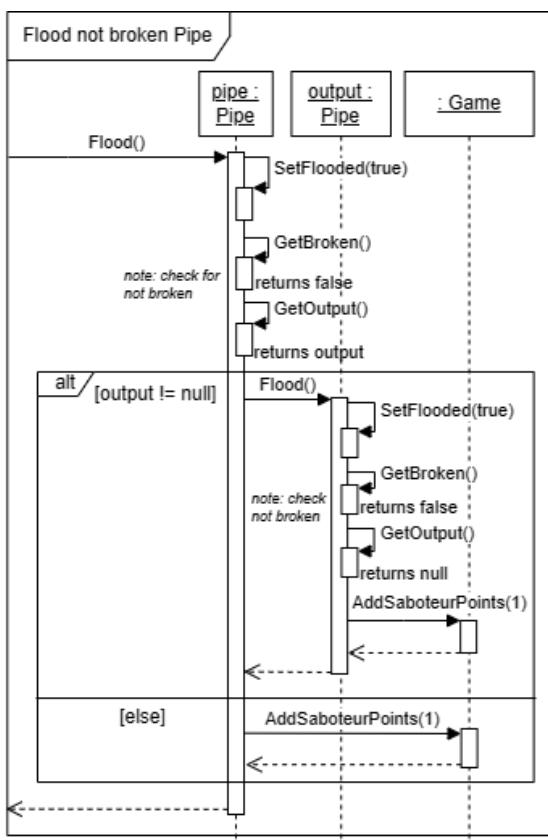
5.3.20 Place Pump when placeable



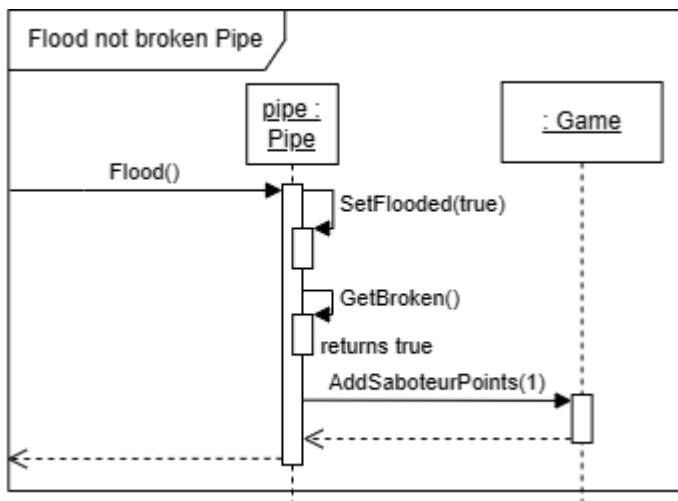
5.3.21 Place Pump when not placeable



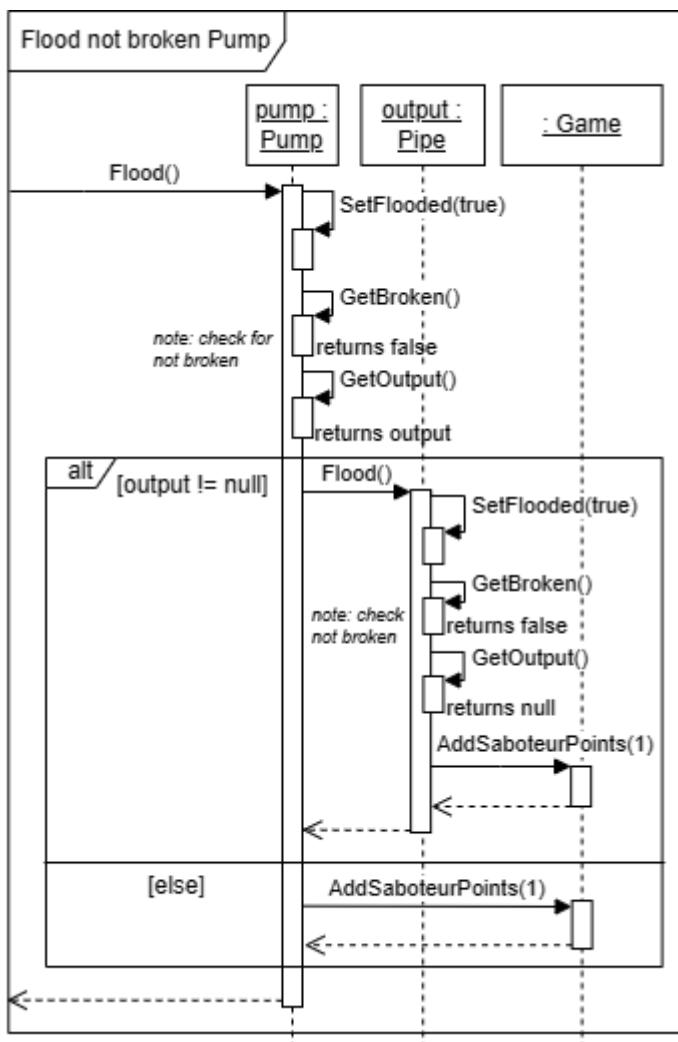
5.3.22 Flood not broken Pipe



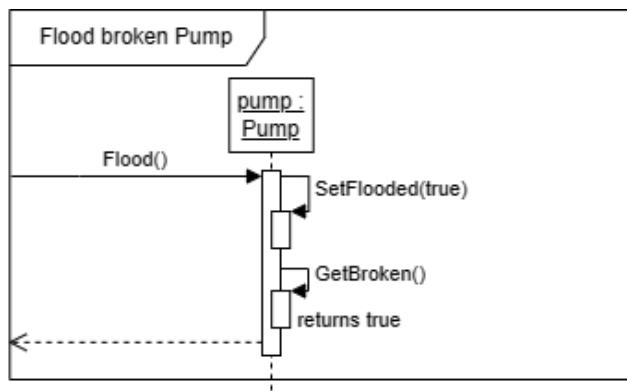
5.3.23 Flood broken Pipe



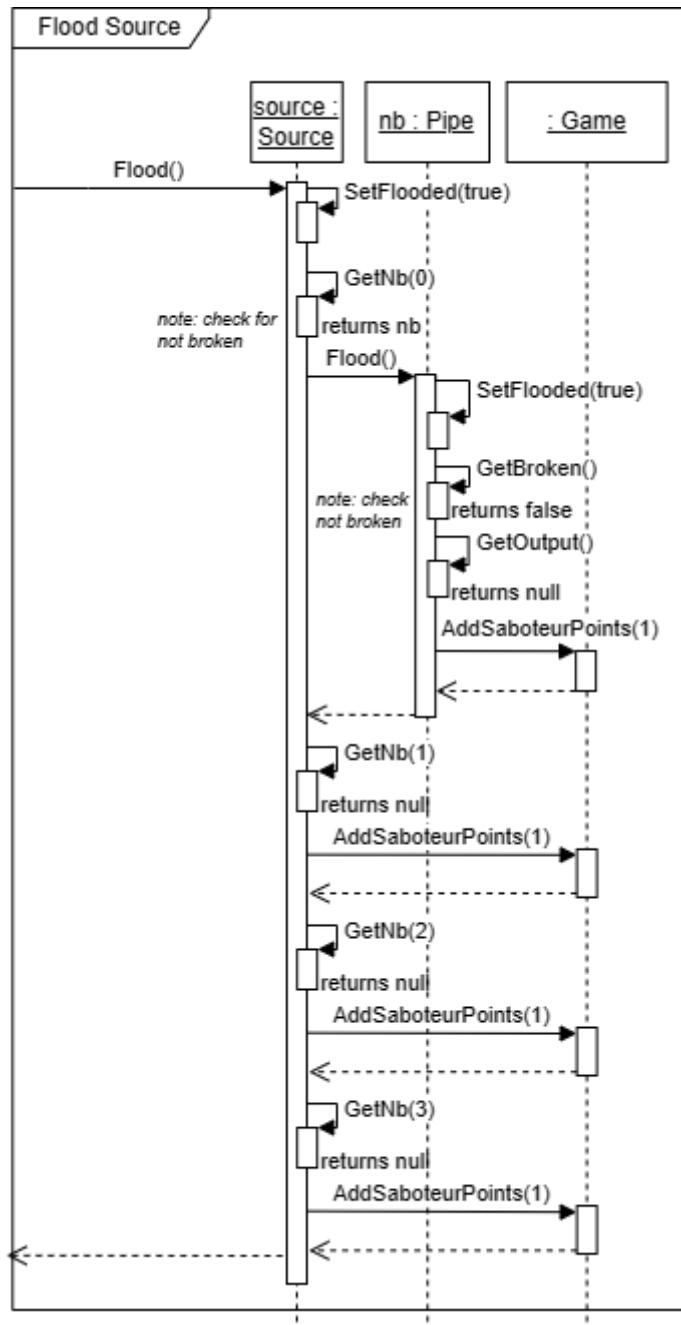
5.3.24 Flood not broken Pump



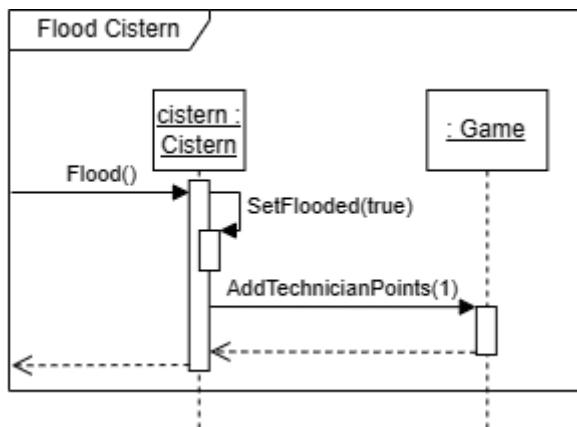
5.3.25 Flood broken Pump



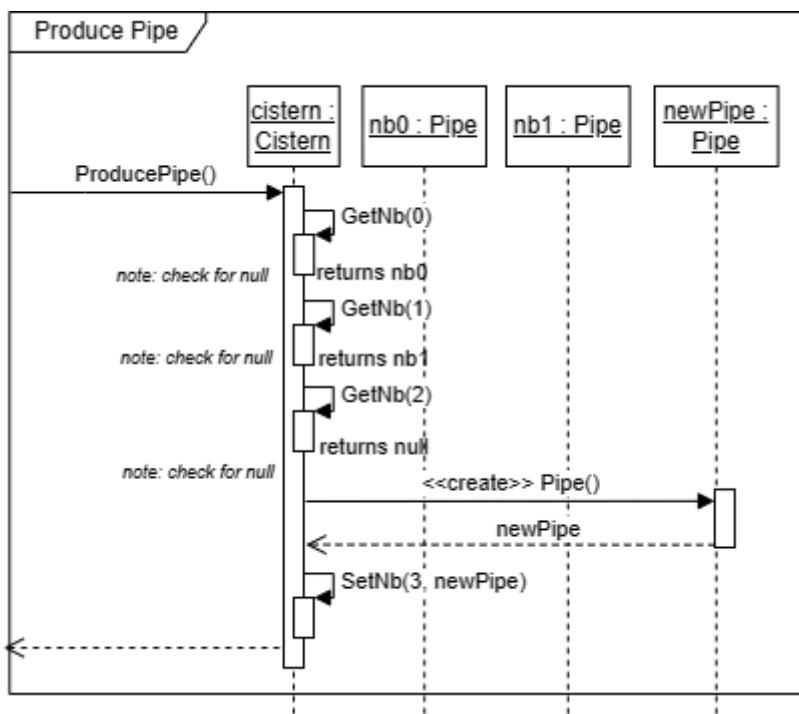
5.3.26 Flood Source



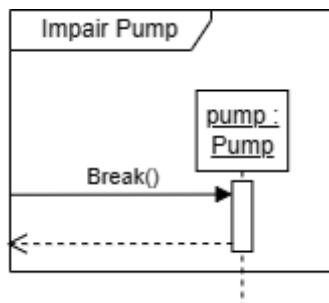
5.3.27 Flood Cistern



5.3.28 Produce Pipe



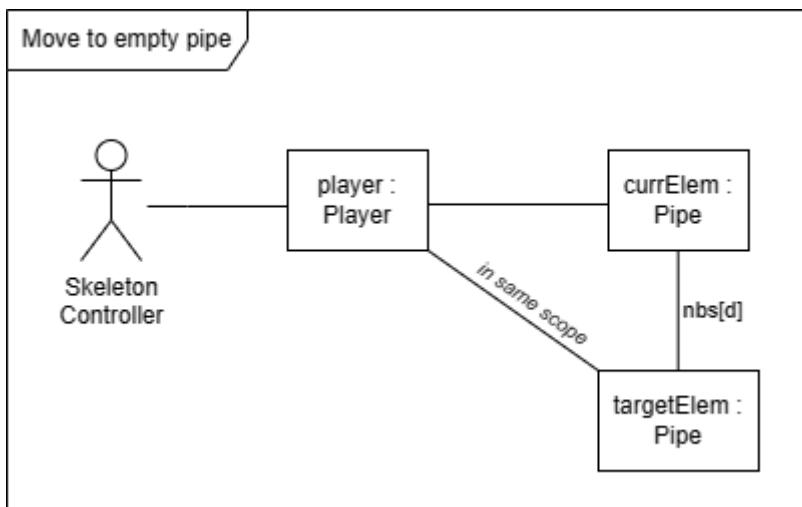
5.3.29 Impair Pump



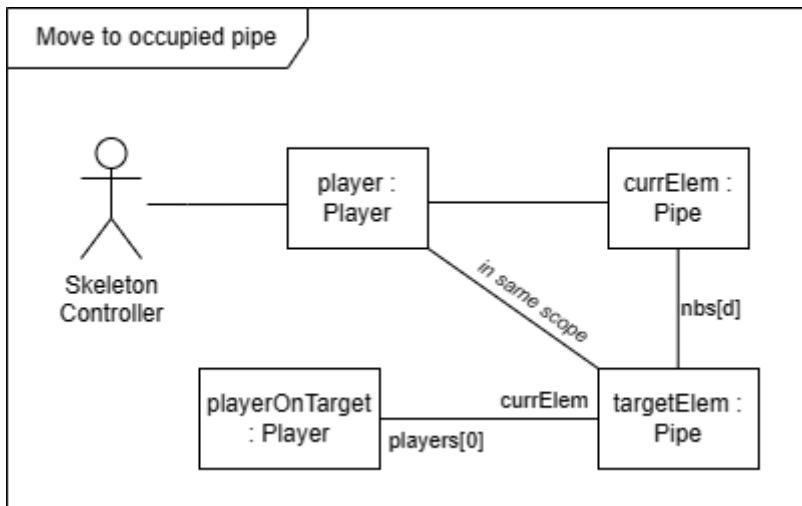
5.4 Kommunikációs diagramok

- A use-case inicializált állapotában mutatja az objektumokat.
- Ha nincs kapcsolat, és a szekvencián mégis kommunikálnak, akkor az inicializált állapotban még ténylegesen nincsen kapcsolat. Paraméterként fogja kapni. De ezt megjegyzésekkel jelöltök, ahol csak tudtuk.

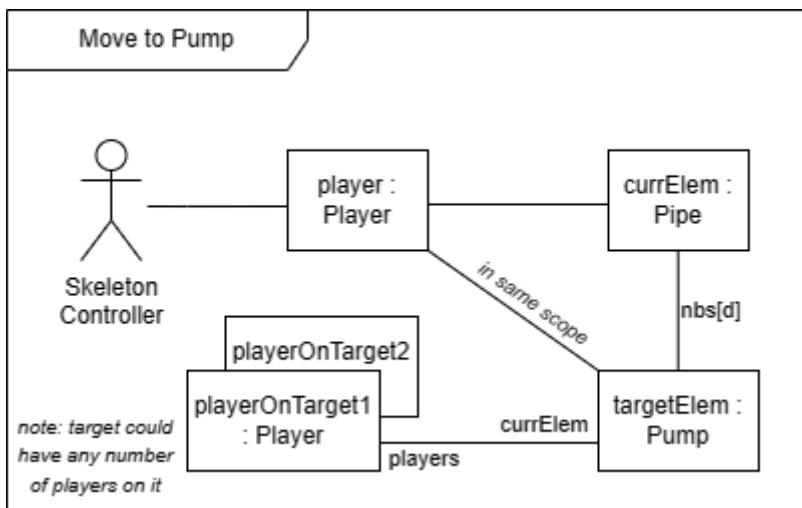
5.4.1 Move to empty Pipe



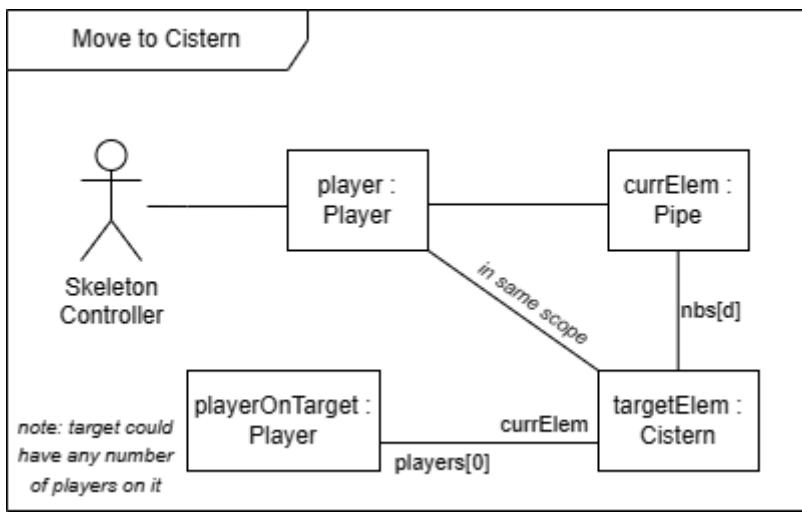
5.4.2 Move to occupied Pipe



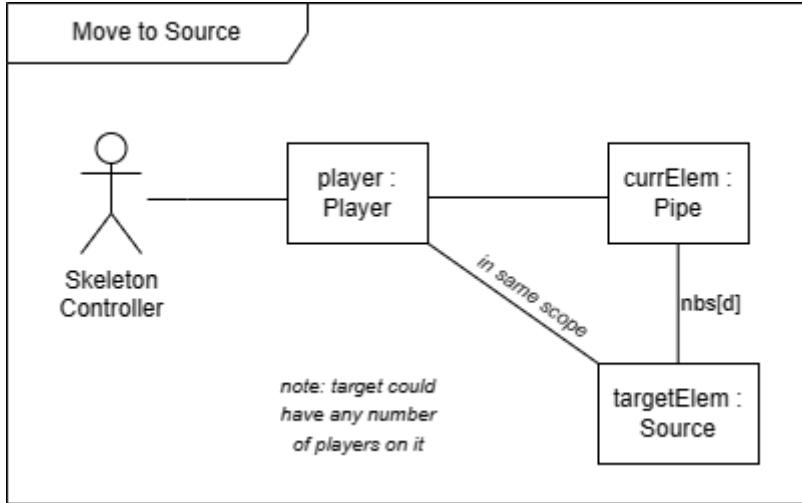
5.4.3 Move to Pump



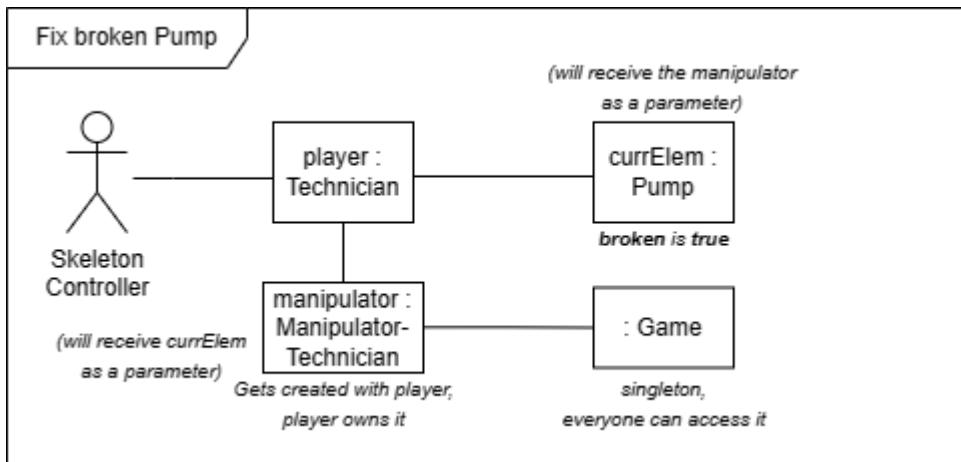
5.4.4 Move to Cistern



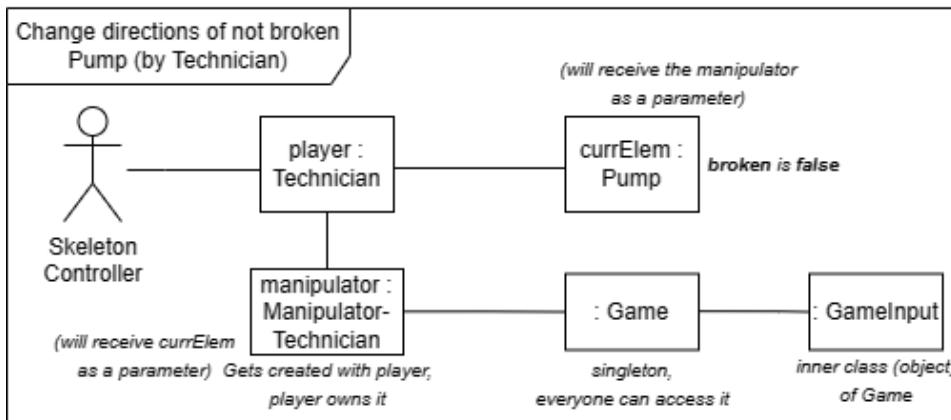
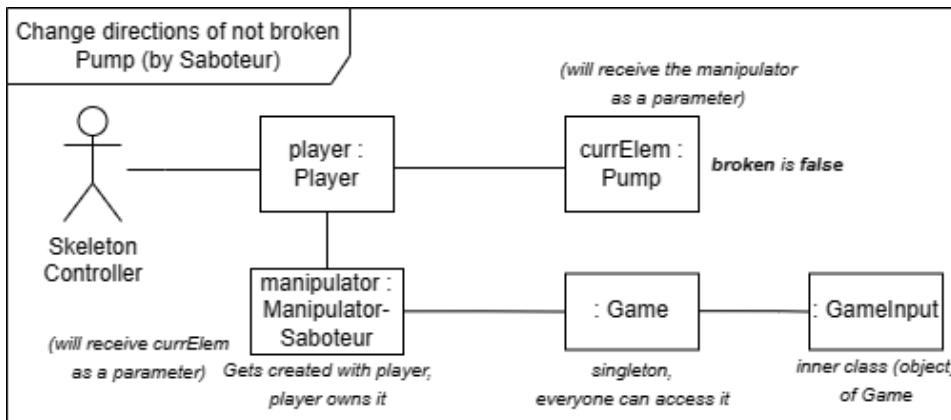
5.4.5 Move to Source



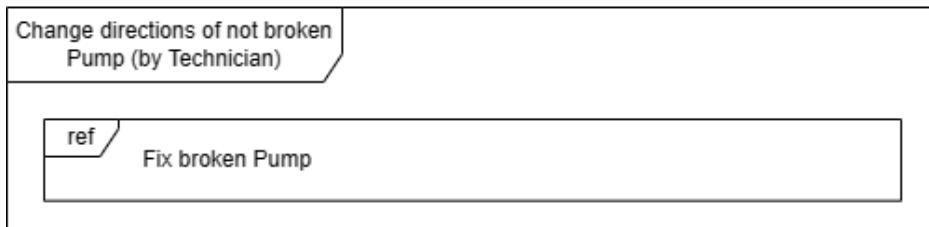
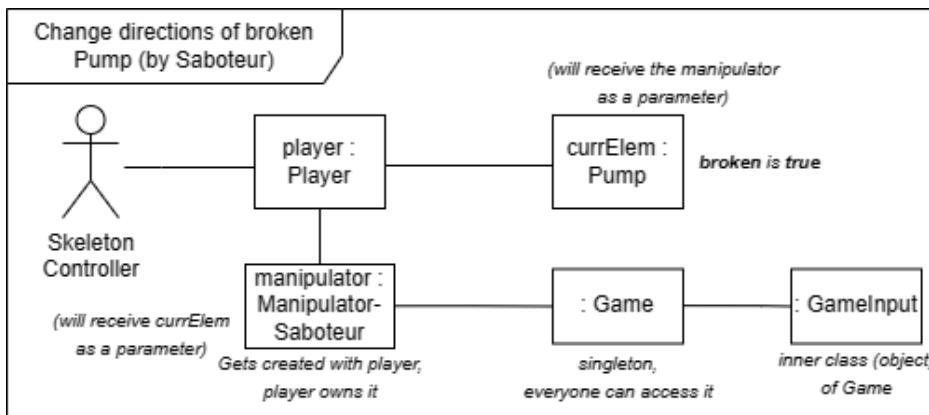
5.4.6 Fix broken Pump



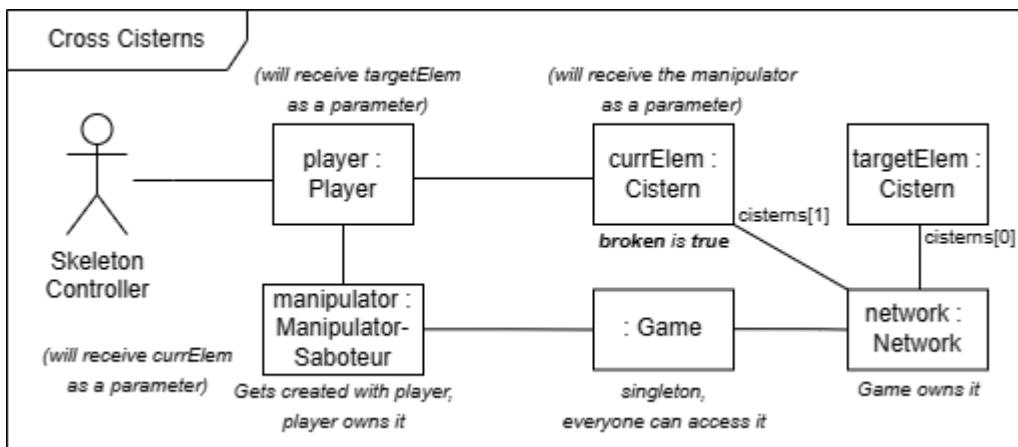
5.4.7 Change directions of not broken Pump



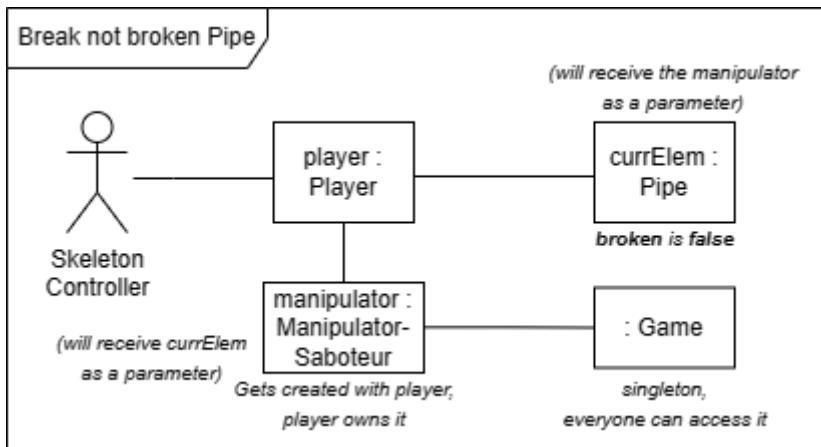
5.4.8 Change directions of broken Pump



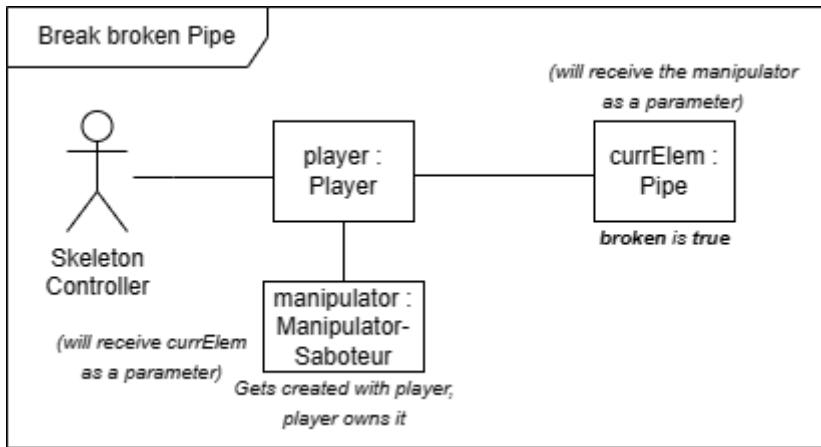
5.4.9 Cross Cisterns



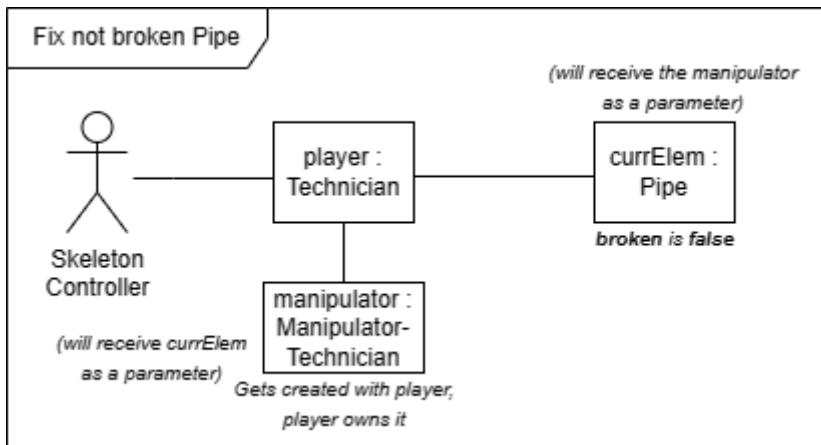
5.4.10 Break not broken Pipe



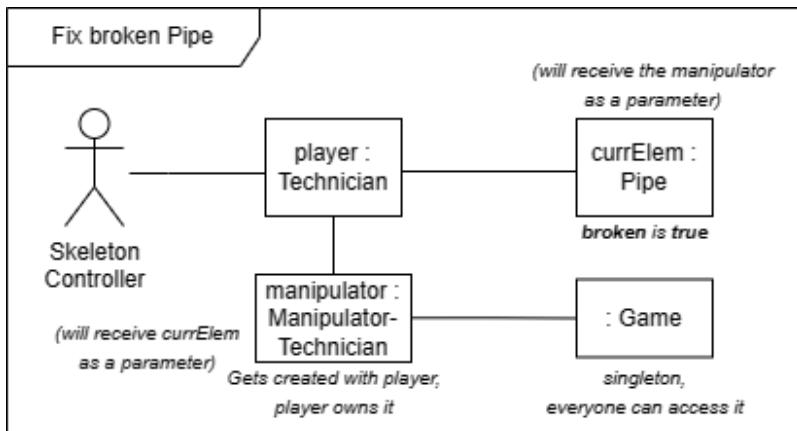
5.4.11 Break broken Pipe



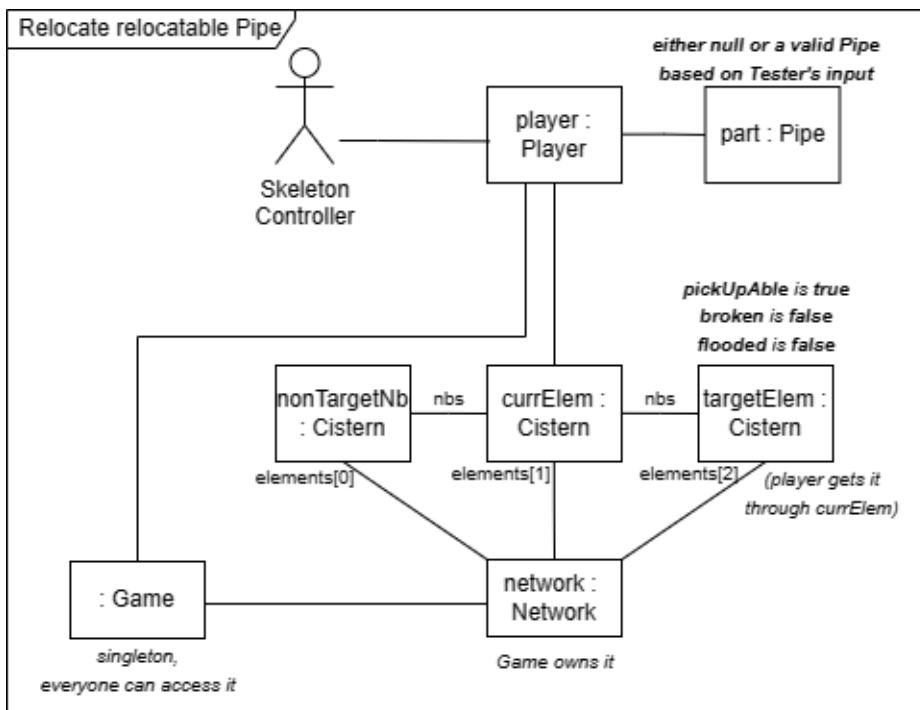
5.4.12 Fix not broken Pipe



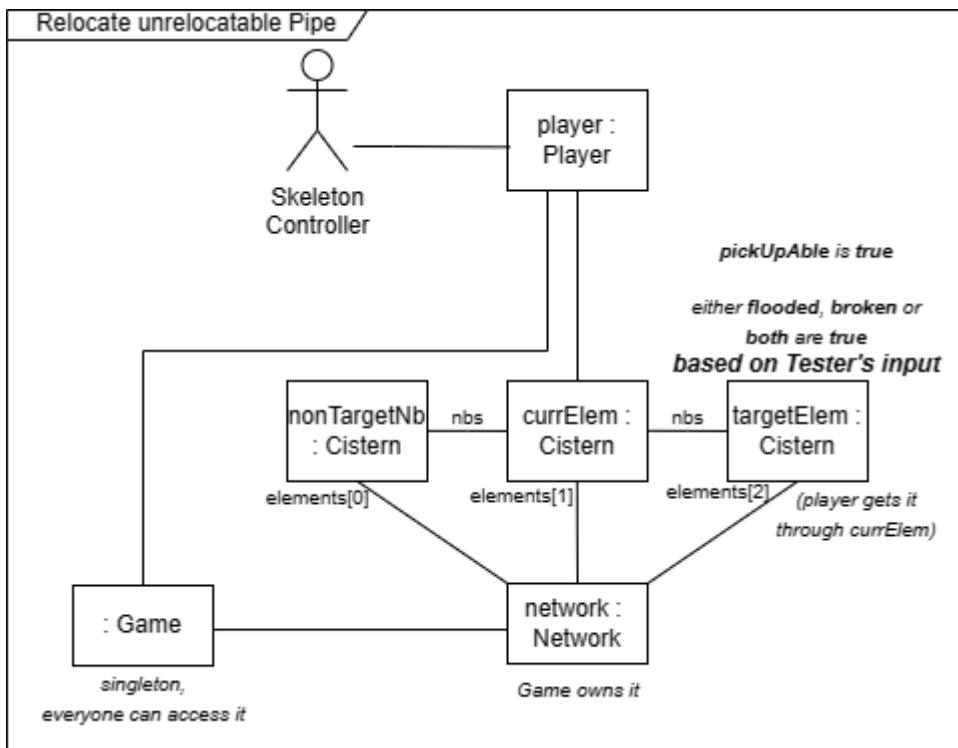
5.4.13 Fix broken Pipe



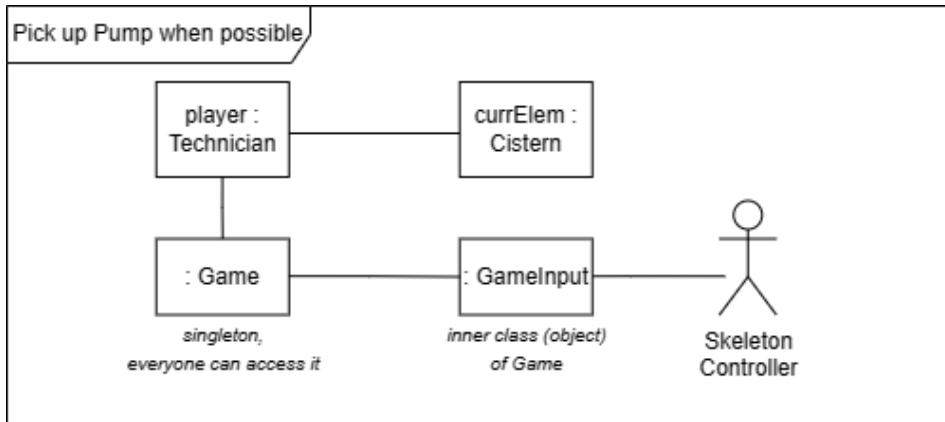
5.4.14 Relocate relocatable Pipe



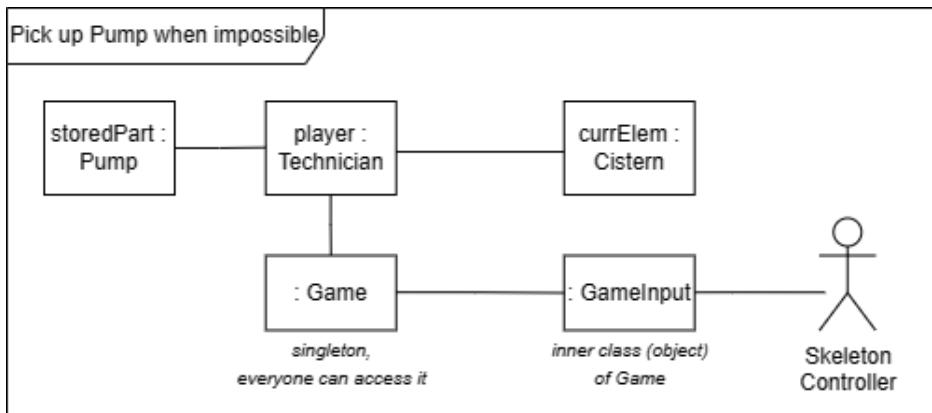
5.4.15 Relocate unrelocatable Element



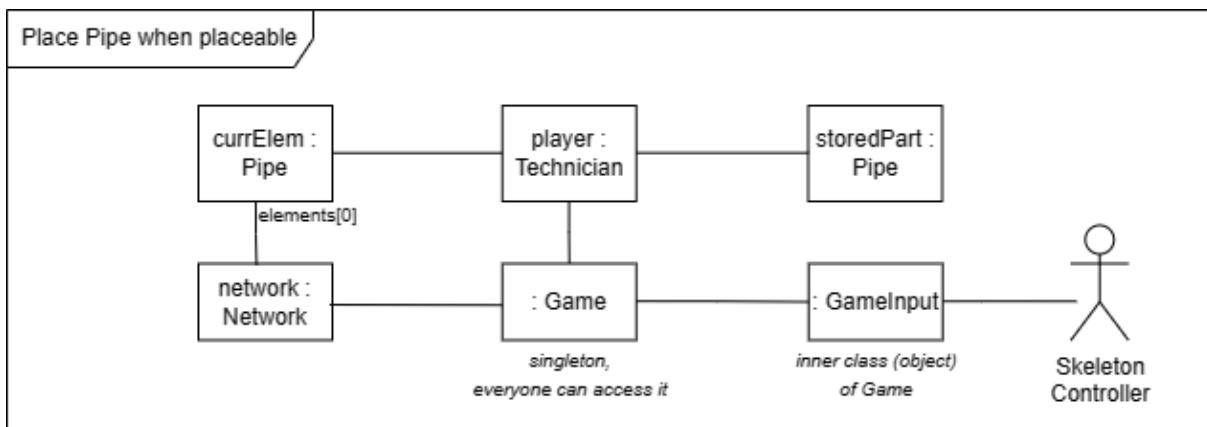
5.4.16 Pick up Pump when possible



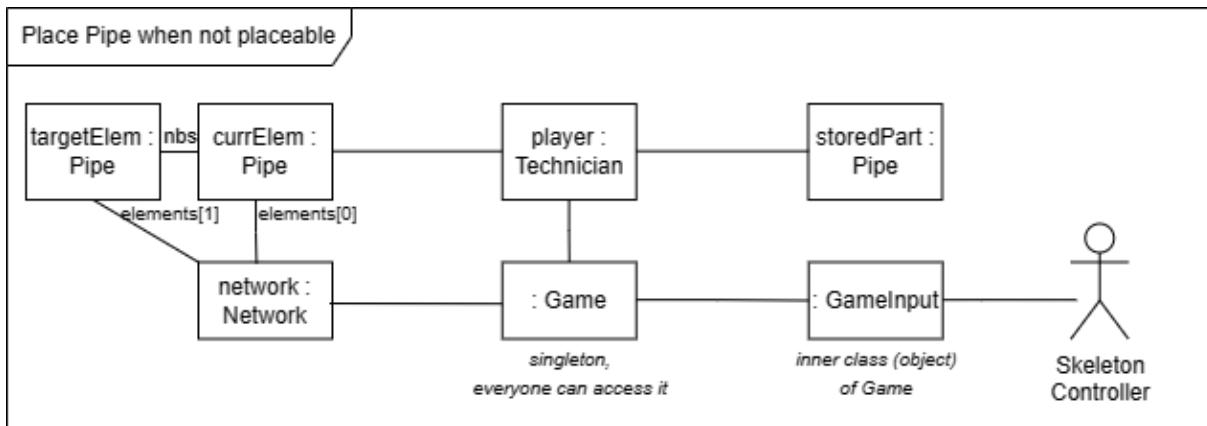
5.4.17 Pick up Pump when impossible



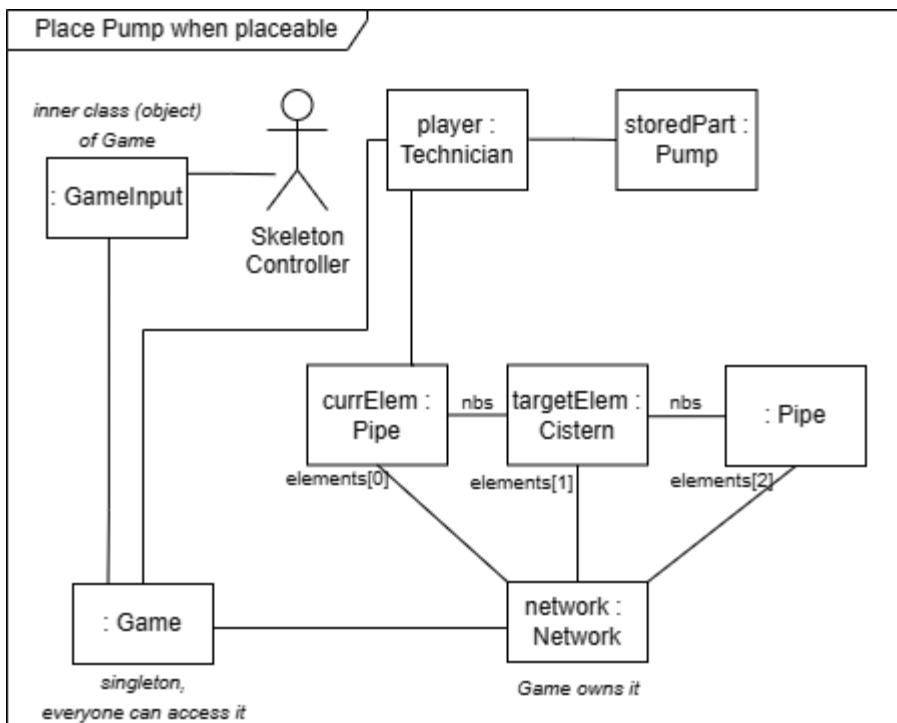
5.4.18 Place Pipe when placeable



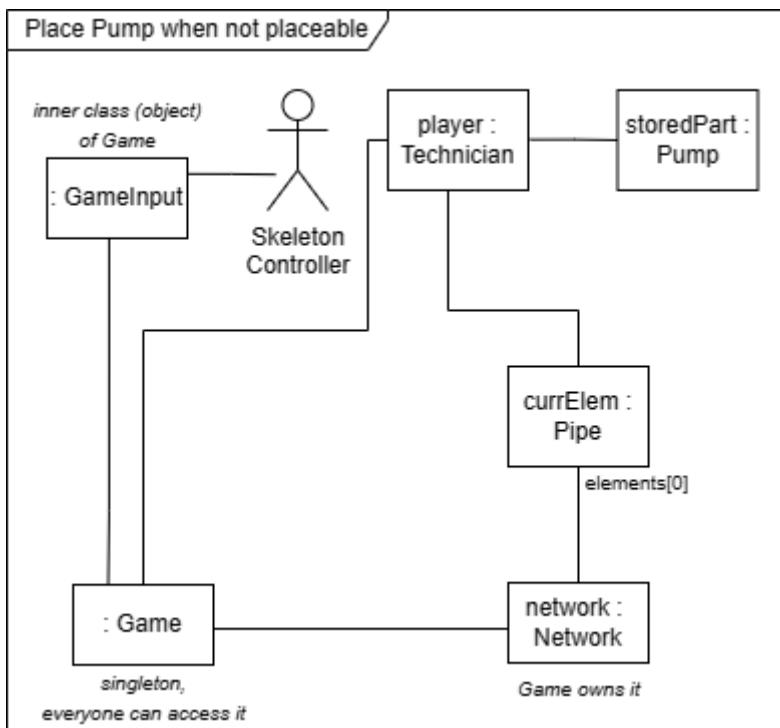
5.4.19 Place Pipe when not placeable



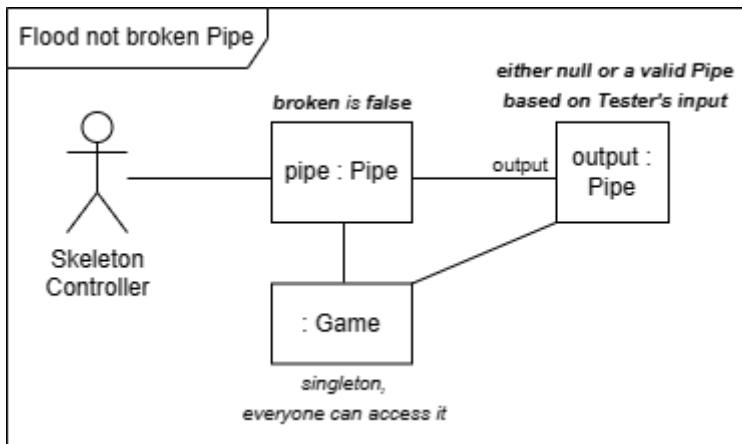
5.4.20 Place Pump when placeable



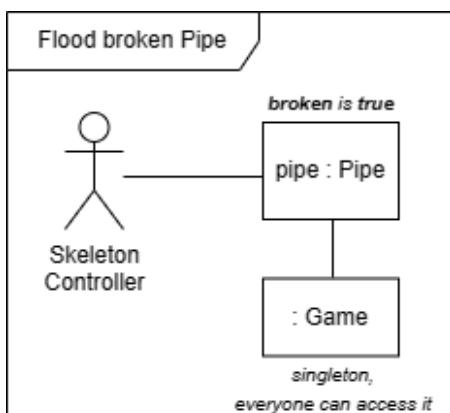
5.4.21 Place Pump when not placeable



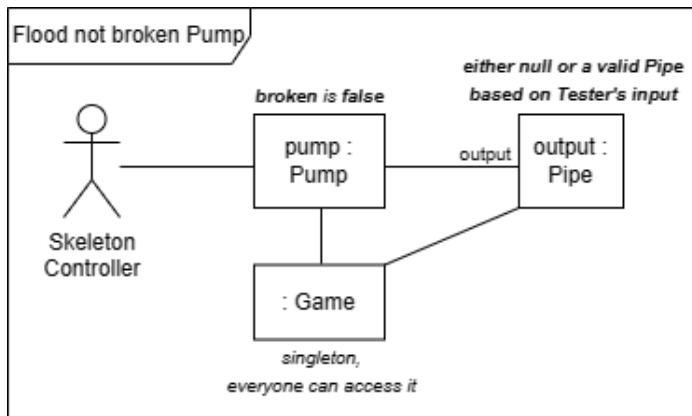
5.4.22 Flood not broken Pipe



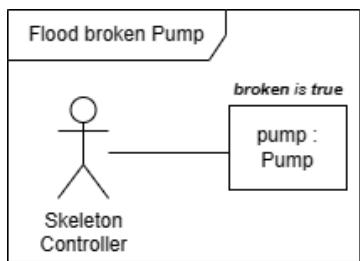
5.4.23 Flood broken Pipe



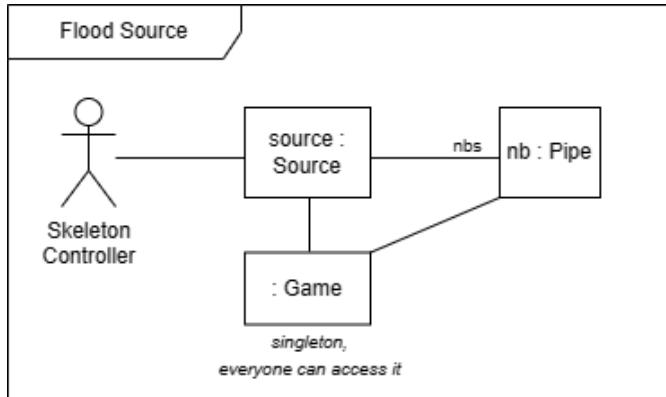
5.4.24 Flood not broken Pump



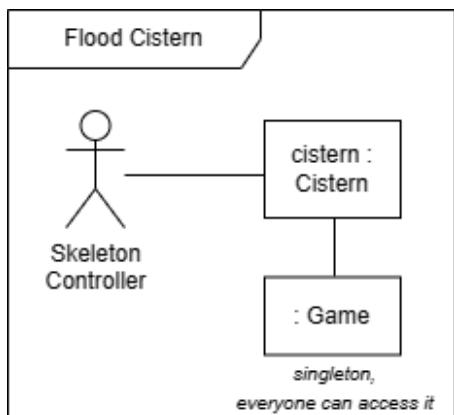
5.4.25 Flood broken Pump

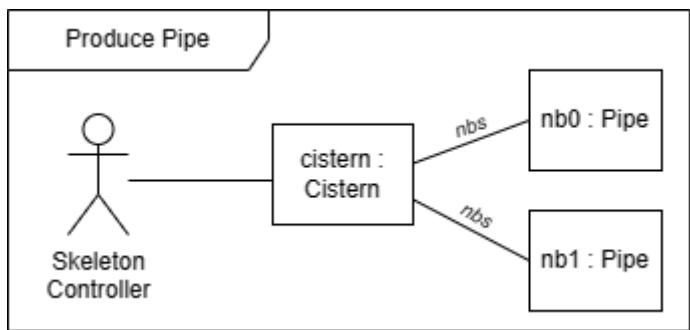
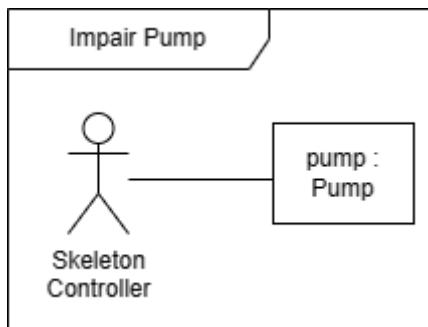


5.4.26 Flood Source



5.4.27 Flood Cistern



5.4.28 Produce Pipe**5.4.29 Impair Pump**

Napló

Kezdet	Időtartam	Résznevők	Leírás
2023.03.29. 19:00	1 óra	Tepliczky Mizser Váradi Sasvári Fekete	<p>Értekezlet. Döntés: Az eheti feladatokat kollaboratív módon, specifikus felosztás nélkül végezzük.</p> <hr/> <p>Use-case-k Határidő: 04.02. 00:00 Elkészült:</p> <p>Szekvencia diagramok Határidő: 04.03. 00:00 Elkészült: 04.01. 21:00</p> <p>Kommunikációs diagramok Határidő: 04.03. 00:00 Elkészült: 04.02. 19:00</p> <p>Teljes dokumentum, formázással Határidő: 04.03 12:00 Elkészült: 04.03 10:00</p>
2023.03.31. 15:30	4 óra	Mizser	Use-case-k összegyűjtése, vázlatos leírások
2023.03.31. 15:30	3 óra	Sasvári	Use-case-k összegyűjtése, vázlatos leírások
2023.03.31. 16:00	3,5 óra	Tepliczky Fekete	Use-case-k összegyűjtése, vázlatos leírások
2023.04.01. 10:30	7 óra	Tepliczky Fekete	Use case leírások kibövitése. Szekvencia diagramok elkezdése (5.3.1 – 5.3.7)
2023.04.01. 12:00	5 óra	Váradi	Szekvencia diagramok folytatása (5.3.7 – 17), korábbiak átnézése
2023.04.01. 14:00	7 óra	Mizser	Use case kapcsolatok, Use case-k véglegesítése Szekvencia diagramok befejezése (5.3.13-29), Kommunikációs diagramok létrehozása (5.4.1 – 5.4.8)
2023.04.01. 18:00	1 óra	Váradi	Kommunikációs diagramok (5.4.9-14)
2023.04.02. 13:00	3 óra	Mizser	Szekvencia- és komm. diagram javítások, Kommunikációs diagramok kiegészítése (5.4.15 – 5.4.21),

5. Szkeleton tervezése

runtime_error

2023.04.02. 15:00	1 óra	Tepliczky Fekete	Use case Diagram (5.1.1)
2023.04.02. 16:00	5 óra	Sasvári	Kommunikációs diagramok befejezése (5.4.22-29), 5.3 és 5.4 egyes részeinek ellenőrzése és javítása. A szkeleton kezelői felületének terve, dialógusok (5.2)
2023.04.03. 08:00	3 óra	Váradi	Dokumentum szerkesztése, diagramok beillesztése (64 db)