

## 2. Követelmény, projekt, funkcionalitás

### 2.1 Bevezetés

#### 2.1.1 Cél

A dokumentum egyfelől definiálja a projekt kereteit, másfelől leírja, hogy a megrendelő mit vár eredményként, mindezt olyan formátumban, hogy a vevő és a szállító is tudja hasznosítani. Ezen kívül ismertet egy ütemtervet, amelyet a szállító csapat követni fog.

#### 2.1.2 Szakterület

A szoftver elsősorban a játékiparban használható. A cél egy olyan termék elkészítése, mely könnyen tanulható, kiegynézőszerű a két csapat szempontjából és nem utolsó sorban szórakoztatónak és hibamentes.

#### 2.1.3 Definíciók, rövidítések

JVM	Java Virtual Machine: Java bájtkód futtatására használt virtuális gép.
Use-case	Használati eset (hogyan fogják a felhasználók használni a programot)
IDE	Integrated Development Environment: Integrált fejlesztői környezet

#### 2.1.4 Hivatkozások

<https://www.jetbrains.com/idea/>

<https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html>

<https://docs.oracle.com/en/database/oracle/oracle-database/21/jjdev/Oracle-JVM-overview.html#GUID-391B265A-D656-4589-A4F5-C4F801340886>

#### 2.1.5 Összefoglalás

Fontos része a dokumentumnak a feladat egységesítése, amit a fejlesztői csapat közösen alkot meg és a 2.2-es szakaszban rögzít. Ez magában foglalja a program funkcióit, a megvalósítás kereteit, felhasználóit, az esetleges korlátozásokat és a külső elemek kapcsolatait a dokumentumhoz. A 2.3 szakasz a program különböző (funkcionális, erőforrásokkal, átadással kapcsolatos) követelményeiről szól. A 2.4-es szakasz a use-case-ekkel foglalkozik, ezeket diagramon is szemlélteti. Fontos, hogy a dokumentumban használt idegen és ritkán használt szavak jelentéseit egységesen rögzítsük, erre szolgál a 2.5-ös szakasz. A 2.6-os szakasz a projekt tervét (határidőket és az ezekhez kapcsolódó feladatokat) írja le. Végül a 2.7-es szakasz arra szolgál, hogy áttekintő képet adjon a csapat egyes résztvevőinek teljesítményéről a dokumentum elkészítése (és az ehhez kapcsolódó tervezési munkák) során.

## 2.2 Áttekintés

### 2.2.1 Általános áttekintés

Az architekturális kép legfelső szintjén találhatóak a játékszereplők, mint például a szerelők vagy a szabotőrök. Ezen rendszerrel a felhasználó közvetlen kapcsolatban áll, mivel billentyűzettel irányítja a mozgásukat, illetve akciókat teljesíthet velük.

Egy szinttel lejjebb található a statikus rendszer, ami a pályán látható hálószerkezet komponenseiből áll, amit a felhasználó csak közvetlen érhet el a dinamikus rendszeren keresztül. Példa erre, hogy kicserélődik a pumpa bemenete és kimenete. Ezt a tevékenységet a szerelő karakteren keresztül lehet megvalósítani, ha a pumpán áll. Egy hasonló példa hogy

kilyukad az egyik csővezeték. Ezt viszont a szabotőr karakter segítségével érhető el, ha a csővezeték közelében van.

A statikus rendszer kapcsolatban áll a legalsó szinten lévő folyami eloszlási rendszerrel. Ez a rendszer felhasználja a statikus rendszer jelenlegi állapotát és ennek következtében kiszámítja a hálózat folyamát, hogy merre kell haladjon a víz. Ha a statikus rendszer változik (az első bekezdés példájával élve, ha a pumpa bemenete és kimenete megváltozik), akkor ennek következtében frissíti a víz által megtett utat. Ez a rendszer felelős a pontozással is, ha víz egy ciszternába jut akkor növeli a szerelők pontszámát.

## 2.2.2 Funkciók

A program az alábbi szabályokat követő játék megvalósítását végzi. Ehhez használ egy főmenüt, amit a felhasználó megnyitáskor lát és később elérhet a játék közben is. Innen indíthatja el a játékot.

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 rágókötött cső eközben el van zárva. A pumpák véletlen időközönként el tudnak romlani, ilyenkor megszűnik az adott pumpánál a vízáramlás. A pumpák mindegyike rendelkezik egy víztartályal, 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 végük lecsatlakoztatva egy másik aktív elemhez elvihetők és ott felcsatlakoztathatók legyenek. A ciszternáknál 3 körönként 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 csőhálózatot a szerelők tartják karban. Ők javítják meg az elromlott pumpákat, ők állítják át a pumpákat, hogy minden lehetséges módon áthidaljanak a hálózaton, és ha egy cső kilyukad, az ő dolguk a cső megfeszítése is. 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 a ciszternáknál magukhoz tudnak venni új pumpát is, amit egy ép cső közepén tudnak elhelyezni. Egy szerelő azonban egyszerre csak egy pumpát tud magával cipelni. A csövet ehhez ketté kell vágni, és a két végét a pumpához kell csatlakoztatni.

A hálózaton élnek a nomád szabotőrök is, akik a pumpákat tudják átállítani és a csöveket szokták kilyukasztani.

Mivel a sivatag veszélyes hely, a szerelők és a szabotőrök csak a csőhálózaton haladhatnak. A pumpáknál kikerülhetik egymást, de a csöveken már nem tudnak elmeni egymás mellett, egy csövön egyszerre csak egy ember állhat.

A játékot a két csapat legalább 2-2 játékossal játszsa. A szabotőrök dolga, hogy minél több víz folyjon el a lyukakon, a szerelők pedig azon dolgoznak, hogy minél több víz jusson a

ciszternában. Az a csapat nyer, amelyik először eléri a játék befejezéséhez szükséges vízmennyiséget (100 egység).

A játék körök alapján működik. Egy körben egy játékos játszik. minden körben a játékos csinálhat 3 feladatot a következők közül:

- áthelyezheti bármely cső egyik végét bárhova máshova
- átállíthatja egy pumpa bemeneti és/vagy kimeneti csövét
- felvehet egy pumpát egy ciszternától (ha szerelő játékos és nincs nála pumpa)
- lerakhat egy pumpát egy ép csőre (ha szerelő játékos és van nála pumpa) és beállíthatja, hogy merre pumpáljon
- megfoltozhat egy csövet (ha szerelő játékos)
- megjavíthat egy pumpát (ha szerelő játékos)
- átkötheti bármely cső egyik végét egy másik helyre (ha szerelő játékos)
- kilyukaszthat egy csövet (ha szabotőr játékos)

A kör végén a pumpák egyet pumpálnak. Azok után a csövek után, amelyek lyukasak, vagy az egyik végük nincs csatlakoztatva pumpához vagy ciszternához és van bennük víz, a szabotőr játékosok kapnak pontot. A lyukas csövekből is ilyenkor folyik ki a víz. Azok után a csövek után, amelyek ciszternákba mennek és van bennük víz, a szerelő játékosok kapnak pontot.

### 2.2.3 Felhasználók

*A szoftver felhasználója egy 6 évnél idősebb személy.*

### 2.2.4 Korlátozások

*Előfeltétele a szoftver használatának egy Windows vagy Unix operációs rendszerrel futó számítógép, illetve a szoftver által szükségeltetett tárhely, ahogy le lehet telepíteni a programot.*

### 2.2.5 Feltételezések, kapcsolatok

IntelliJ IDEA A fejlesztői környezet, amit a csapat használni tervez

Java SWING A grafikus megjelenítésre használt könyvtár leírása

JVM A program futtatásához szükséges program leírása

## 2.3 Követelmények

### 2.3.1 Funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Use-case	Koment
F01	A program elindításakor a menü fogadja a játékost	A játékot elindítjuk és megfigyeljük	Opcionális	Csapat	képernyőt néz	

F02	<i>A menüben a játékos új játékot indíthat vagy kiléphet</i>	A menüt megnyitjuk és megfigyeljük	Opcionális	Csapat	képernyőt néz	
F03	<i>Játék indítása esetén meg kell adni ki játszik melyik csapatban, ezután a játék képernyő fogadja a játékost</i>	<i>Elinítjuk a menüből a játékot és megfigyeljük</i>	Opcionális	Csapat	képernyőt néz	
F04	<i>A játék képernyőn a menü gombot megnyomva hozható elő a menü</i>	<i>Megnyomjuk a menü gombot és megfigyeljük XD</i>	Opcionális	Csapat	képernyőt néz	
F05	<i>Ha játék közben előhozza a játékos a menüt, akkor kiléphet a programból, folytathatja az aktuális játékot, vagy új játékot indíthat</i>	<i>Játék közben megnyomjuk a menü gombot és megfigyeljük</i>	Opcionális	Csapat	képernyőt néz	
F06	<i>A játékosok a játék képernyőn végezhetnek feladatokat</i>	<i>A játék képernyőn irányítjuk a karaktert és végezzük a feladatokat</i>	Alapvető	Csapat	szabályok ellenőrzése	
F07	<i>Minden körben egy játékos végezhet feladatokat</i>	<i>Egy körben több játékos is megpróbál feladatot végezni</i>	Alapvető	Csapat	szabályok ellenőrzése	
F08	<i>Egy játékos a körében három feladatot végezhet</i>	<i>A játékos megpróbál 3-nál több feladatot végezni</i>	Fontos	Csapat	szabályok ellenőrzése	
F09	<i>Ha a játékos elvégezte a három feladatát vagy a kör vége gombra nyom, akkor vége a körének és a következő játékos következik</i>	<i>Gombot megnyomjuk és megfigyeljük</i>	Alapvető	Csapat	szabályok ellenőrzése	
F10	<i>A szerelő játékosok végezhető feladatai:</i> <ul style="list-style-type: none"><li>• cső átirányítása</li><li>• pumpa be-/kimenet átirányítása</li><li>• pumpa felvétel</li><li>• pumpa lerakás</li><li>• lyukas cső megfoltozása</li><li>• elromlott pumpa megjavítása</li></ul>	<i>Egy szerelő játékos a felsorolt műveleteket megpróbálja elvégezni</i>	Alapvető	Feladatleírás	szabályok ellenőrzése, pumpát szerel, csövet szerel, pumpát felvesz	
F11	<i>A szabotőr játékosok végezhető feladatai:</i>	<i>Egy szabotőr játékos a</i>	Alapvető	Feladatleírás	szabályok ellenőrzése,	

	<ul style="list-style-type: none"> <li>• pumpa be-/kimenet átirányítása</li> <li>• cső kilyukasztása</li> </ul>	felsorolt műveleteket megpróbálja elvégezni			csövet kilyukaszt	
F12	A cső egyik végének átirányítása egy elvégzett feladat	Egy játékos elvégzi egy cső átirányítását és megfigyeli hány feladatot végezhet még	Fontos	Csapat	szabályok ellenőrzése	
F13	Akárhonnan át lehet irányítani egy csövet	Egy játékos megpróbál átirányítani csövet különböző helyéről	Opcionális	Csapat	szabályok ellenőrzése	
F14	A pumpa bemenetének vagy kimenetének átállítása egy elvégzett feladat	Egy játékos elvégzi egy pumpa átállítását és megfigyeli hány feladatot végezhet még	Fontos	Csapat	szabályok ellenőrzése, pumpát átállít	
F15	Akkor állítható át a pumpa be-/kimenete, ha a játékos rajta áll	Egy játékos megpróbálja átállítani a pumpát különböző helyéről	Fontos	Csapat	szabályok ellenőrzése, pumpát átállít	
F16	Pumpát a cisternákról lehet felvenni	Egy játékos a cisternára lép és felveszi a pumpát	Fontos	Feladatleírás	szabályok ellenőrzése, pumpát felvesz	
F17	Egy játékos egyszerre egy pumpát tarthat magánál	Egy játékos megpróbál több pumpát felvenni	Fontos	Csapat	szabályok ellenőrzése	
F18	Pumpát egy csőre lehet lerakni, ekkor a korábbi csőből két cső lesz, amik az újonnan lerakott pumpához csatlakoznak	Egy játékos megpróbál lerakni egy pumpát csőre, illetve más hova, de ott nem sikerül	Alapvető	Feladatleírás	szabályok ellenőrzése	
F19	Lyukas csövet akkor javíthat meg egy szerelő, ha rajta áll	Egy szerelő játékos megpróbál megjavítani egy csövet, úgy, hogy nem rajta áll	Fontos	Csapat	szabályok ellenőrzése, csövet szerel	

F20	<i>Ép csövet akkor lyukaszthat ki egy szabotőr, ha rajta áll</i>	<i>Egy szabotőr játékos megpróbál kilyukasztani úgy, hogy nem rajta áll</i>	Fontos	Csapat	<i>szabályok ellenőrzése, csövet kilyukaszt</i>	
F21	<i>Egy elromlott pumpát akkor javíthat meg egy szerelő, ha rajta áll</i>	<i>Egy szerelő játékos megpróbál megjavítani egy pumpát, úgy, hogy nem rajta áll</i>	Fontos	Csapat	<i>szabályok ellenőrzése, pumpát szerel</i>	
F22	<i>Ha egy cső lyukas, vagy az egyik vége nincsen pumpára kapcsolva, akkor minden körben kifolyik egy adag víz és a szabotőrök pontot kapnak</i>	<i>Egy csőön előidézzük a helyzetet és megfigyeljük kap-e pontot a szabotőr csapat</i>	Alapvető	Feladatleírás	<i>szabályok ellenőrzése</i>	
F23	<i>Ha egy cisternába egy bemeneten víz folyik, akkor azért az egy bemenetért a szerelők körönként kapnak egy pontot, több bemenetért több pontot</i>	<i>Egy cisternára vizet kapcsolunk és megfigyeljük kap-e pontot a szerelő csapat</i>	Alapvető	Feladatleírás	<i>szabályok ellenőrzése</i>	
F24	<i>Az a csapat nyer, amelyik előbb éri el a 100 pontot</i>	<i>Összegyűjtünk egy csapattal 100 pontot és megfigyeljük, hogy leáll-e a játék</i>	Fontos	Csapat	<i>szabályok ellenőrzése</i>	
F25	<i>A pumpák random időközönként maguktól elromlanak</i>	<i>Várunk a pumpák elromlására</i>	Alapvető	Feladatleírás	<i>Pumpát elront</i>	
F26	<i>A játékosok csak csövekre, pumpáakra és cisternáakra léphetnek</i>	<i>Egy játékos megpróbál másra is lépni</i>	Alapvető	Feladatleírás		
F27						

### 2.3.2 Erőforrásokkal kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
E01	<i>A fejlesztési környezetnek támogatnia kell az <a href="#">IntelliJ IDEA</a> fejlesztő környezetet</i>	<i>Telepítjük a IDE-t és megpróbáljuk buildelni a projekt</i>	Fontos	Csapat	

		<i>legfrissebb verzióját</i>			
E02	<i>A futtatási környezetnek támogatnia kell a <a href="#">Java Swing</a> alapú grafikus megjelenést</i>	<i>Megpróbáljuk futtatni a projekt legfrissebb grafikus verzióját</i>	<i>Alapvető</i>	<i>Feladat</i>	
E03	<i>A futtatási környezetnek rendelkeznie kell <a href="#">JVM</a>-mel és legyen képes Java kódot futtatni</i>	<i>A terminálba beírjuk, hogy 'java -version' és ha a Java nyelv és a JVM támogatva van, akkor visszakapjuk a rendelkezésre álló java környezet verziósztmát</i>	<i>Alapvető</i>	<i>Feladat</i>	

### 2.3.3 Átadással kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
A01	<i>Az átadott program egy futtatható állományként indítható</i>	<i>A program futtatható állományát a desktopról vagy a terminálból megpróbáljuk futtatni</i>	<i>Fontos</i>	<i>Csapat</i>	
A02					

### 2.3.4 Egyéb nem funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
N01	<i>A program váratlan hibák nélkül zökkenőmentesen fut, amíg ki nem lépünk</i>	<i>A program minden funkcióját kipróbáljuk</i>	<i>Alapvető</i>	<i>Csapat</i>	
N02	<i>A játék felülete reszponzív, a képfrissítési ráta 30 feletti</i>	<i>A program minden funkcióját kipróbáljuk és pár kört játszunk</i>	<i>Fontos</i>	<i>Csapat</i>	

N03					
-----	--	--	--	--	--

## 2.4 Lényeges use-case-ek

### 2.4.1 Use-case leírások

<b>Use-case neve</b>	<b>Mozog</b>
<b>Rövid leírás</b>	A játékos a csövek és pumpák hálózatán keresztül mozog.
<b>Aktorok</b>	Játékos
<b>Forgatókönyv</b>	1.A játékos a karakterét valamely szomszédos pálya elemre mozgathatja.
<b>Alternatív forgatókönyv</b>	1.1.A játékos csőről pumpára lép, a csövet szabaddá teszi.
<b>Alternatív forgatókönyv</b>	1.2.A játékos pumpáról csőre lép, a csövet elfoglalja.

■

<b>Use-case neve</b>	<b>Pályát néz</b>
<b>Rövid leírás</b>	A játékos megtekinti a képernyőt, vagyis a pályát vagy a menüt.
<b>Aktorok</b>	Játékos
<b>Forgatókönyv</b>	1.A rendszer kirajzolja a képernyő aktuális állapotát. 2.A játékos megtekinti a képernyő aktuális állapotát.

<b>Use-case neve</b>	<b>Pumpát átállít</b>
<b>Rövid leírás</b>	A játékos átállítja a pumpát, ezzel manipulálja a víz folyási irányát.
<b>Aktorok</b>	Játékos
<b>Forgatókönyv</b>	1.A játékos átállítja a pumpát, ezzel manipulálja a víz folyási irányát.

<b>Use-case neve</b>	<b>Pumpa tönkremegy</b>
<b>Rövid leírás</b>	A controller véletlenszerűen elrontja a pumpákat.
<b>Aktorok</b>	Controller
<b>Forgatókönyv</b>	1.A controller véletlenszerűen elrontja a pumpákat.

<b>Use-case neve</b>	<b>Csövet kilyukaszt</b>
<b>Rövid leírás</b>	A szabotörök kilyukaszthatják a csöveket.
<b>Aktorok</b>	Szabotör
<b>Forgatókönyv</b>	1.A szabotörök kilyukaszthatják a csöveket.

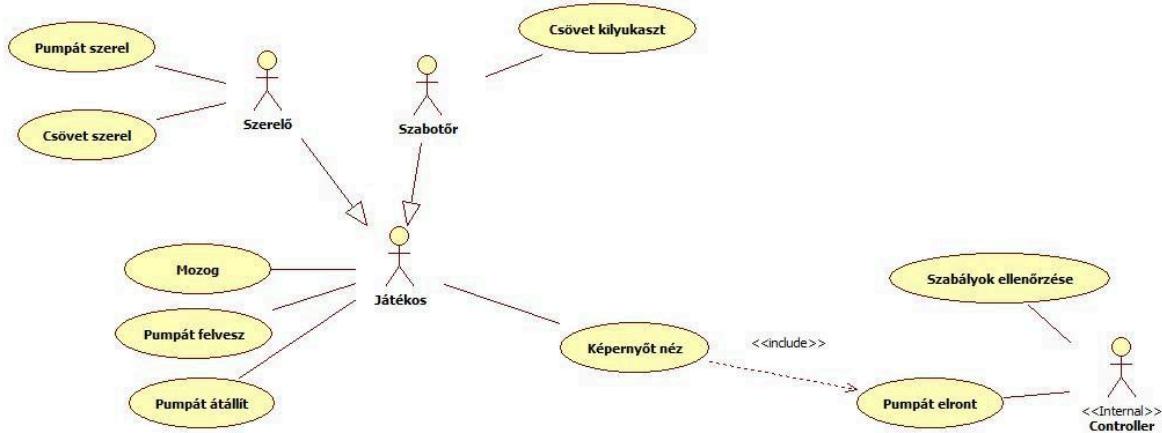
<b>Use-case neve</b>	<b>Csövet szerel</b>
<b>Rövid leírás</b>	A szerelők befoltozhatják a csöveget.
<b>Aktorok</b>	Szerelő
<b>Forgatókönyv</b>	1.A szerelők befoltozhatják a csöveget.

<b>Use-case neve</b>	<b>Pumpát szerel</b>
<b>Rövid leírás</b>	A szerelők megjavíthatják a pumpákat.
<b>Aktorok</b>	Szerelő
<b>Forgatókönyv</b>	1.A szerelők megjavíthatják a pumpákat.

<b>Use-case neve</b>	<b>Pumpát felvesz</b>
<b>Rövid leírás</b>	A szerelők magukhoz vesznek egy pumpát a ciszternáknál.
<b>Aktorok</b>	Szerelő
<b>Forgatókönyv</b>	1.A szerelők magukhoz vesznek egy pumpát.

<b>Use-case neve</b>	<b>Szabályok ellenőrzése</b>
<b>Rövid leírás</b>	A controller ellenőrzi a szabályokat, irányítja a játékot.
<b>Aktorok</b>	Controller
<b>Forgatókönyv</b>	1.A controller ellenőrzi a szabályokat, irányítja a játékot.

## 2.4.2 Use-case diagram



## 2.5 Szótár

ciszterna	Azon eleme a játéknak, amely a vízhálózat
-----------	---

	végpontjain lehet. Ide gyűjtik a szerelők a vizet. Egy egységes víz begyűjtése egy pontot jelent a szerelőknek.
cső	Azon eleme a játéknak, amely a vízhálózat egy középső eleme lehet. Egy cső két pumpa között helyezkedik el, hosszuk változó lehet. Folyik át rajta a víz, illetve ezeken tudnak mozogni a játékosok.
cső végének átirányítása	A csöveket át lehet szerelni a vízhálózat másik pontjára úgy, hogy az egyik végét átköti egy szerelő egy másik pumpához vagy ciszternához.
egy adag víz	Az a mértékegység a játékban, ami egy pontot ér.
fejlesztési környezet	Az a program, amiben a játék működéséért felelős kódot meg lehet írni.
foltozás	Egy kilyukadt cső megszerelése, ami után már nem lyukas.
futtatási környezet	Egy olyan informatikai eszköz, melyen a programot el lehet indítani.
grafikus megjelenítés	A program azon felülete, melyet a felhasználó vagy játékos szabad szemmel lát.
IntelliJ IDEA	Egy fejlesztési környezet.
JVM	A program futtatásához szükséges elem.
Java	Egy programozási nyelv.
Java Swing	A Java-nak egy kiegészítése, mellyel grafikus megjelenítést lehet létrehozni.
játék képernyő	Azt nevezük így, amikor a felhasználó által szemlélt képernyőn nem a menü jelenik meg, hanem a játék.
képfrissítési ráta	Annak a mértékegysége, hogy a képernyő hányszor frissül egy másodperc alatt.
kód	Az a szöveg, amely egy programozási nyelven íródott, és a program működését írja le.
kör	Egy játékos által végzett játékbeli cselekvések összessége. Egyszerre csak egy játékos végezhet játékbeli cselekvéseket. Ekkor van az örökre.
kör vége gomb	Az a gomb, melynek lenyomásakor egy játékos jelzi, hogy végzett a saját körével, és átadja a cselekvés lehetőségét egy másik játékosnak.
menü	Amikor a felhasználó által megfigyelt képernyőn a játékkal kapcsolatos lehetőségei vannak felsorolva, melyekből választhat.
pont	A játék megnyeréséhez gyűjtendő játékelem. Annak a mértékegysége, amely azt jelzi, hogy melyik csapat milyen közel áll a győzelemhez.

program	Amit a számítógépen el lehet indítani, ami különböző tevékenységeket tesz lehetővé a felhasználó számára a számítógépen.
program futás	A program elindítása.
programozási nyelv	Olyan szavak és kifejezések összessége, melyek bizonyos szintaktikai és szemantikai szabályoknak felelnek meg, és leírható velük egy program működése.
pumpa	Azon eleme a játéknak, amely a vízhálózat egy középső eleme lehet. Csöveket köt össze. A víz megfelelő irányú áramlását biztosítja a játékban.
pumpa bemenet	Az a cső, amelyiken befolyik a pumpába a víz.
pumpa kimenet	Az a cső, amelyiken kifolyik a pumpából a víz.
random	Véletlenszerű.
szabotör	Az a játékos, melynek célja, hogy úgy szerezzen pontokat, hogy a víz kifolyik a vízhálózatból.
szerelő	Az a játékos, melynek célja, hogy úgy szerezzen pontokat, hog a víz befolyik a ciszternákba.

## 2.6 Projekt terv

Feladat	Feladatleírás	Határidő
Követelmény, projekt, funkcionalitás	<i>A projekt általános leírása, követelményeinek és kereteinek definiálása. Megrendelői elvárások és projekt terv dokumentálása.</i>	március. 13. 14:15
Analízis modell (I. változat)	<i>A program struktúrájának megtervezése. Osztályok leírása, osztálydiagram, szekvenciadiagramok és állapotgépek elkészítése.</i>	márc. 20. 14:15
Analízis modell (II. változat)	<i>Az analízis modell első változatának javítása, kiegészítése.</i>	márc. 27. 14:15

<i>Szkeleton tervezése</i>	<i>A programváz tervezése. Főbb osztályok, interfések és metódusok leírása.</i>	ápr. 3. 14:15
<i>Szkeleton elkészítése</i>	<i>A programváz elkészítése a tervezek alapján.</i>	ápr. 17. 14:15
<i>Prototípus koncepciója</i>	<i>A program logikai működésének teljes leírása.</i>	ápr. 24. 14:15
<i>Részletes tervezek</i>	<i>A prototípusban szereplő osztályok, metódusok, attribútumok részletes dokumentálása.</i>	máj. 3. labor
<i>Prototípus elkészítése</i>	<i>A prototípus elkészítése a tervezek alapján.</i>	máj. 8. 14:15
<i>Grafikus változat tervezettsége</i>	<i>A program grafikai interfészének megtervezése.</i>	máj. 15. 14:15
<i>Grafikus változat elkészítése</i>	<i>A grafikus interfész elkészítése a tervezek alapján.</i>	máj 31. labor
<i>Egyesített dokumentáció</i>	<i>A dokumentációk egybegyűjtése és azok véglegesítése.</i>	jún. 2. 14.00

A megfelelő elosztott munkakörnyezet megteremtéséhez szükséges, hogy a programfejlesztő csapat minden tagjának legyen saját számítógépe, melyen olyan operációs rendszer fut, amellyel kielégíthetők a 2.3.2-es fejezetben leírt követelmények.

#### Munkakörnyezet leírása:

- Az értekezletek élőben vagy online formában történnek. Az online “meeting”-ek egy közös Discord szerveren történnek, illetve ugyanitt kerülnek megosztásra fontos információk is.
- A feladatokkal kapcsolatos kérdéseket, segítséggéréseket és általános információk megosztását egy Messenger csoportban végezzük.
- A dokumentációk elkészítését és szerkesztését a Google Docs segítségével csináljuk, melyet egy a csapat minden tagjának megosztott Google Drive-ban gyűjtünk.
- A feladatok kiosztását és monitorozását az Azure DevOps Board-ján végezzük.
- A program forráskódok megosztásához Azure DevOps Git szervert használunk.

## 2.7 Napló

Kezdet	Időtartam	Résznevők	Leírás
2023.03.08 20:00	1 óra	Mali Zavadil Völgyesi Garai Varga	Értekezlet. Döntés: Azure DevOps-t használjuk projekttámogató szoftverként. Google Docs segítségével szerkesztjük a fájlokat egyszerre többen. Discord-ot használunk a kommunikációra.
2023.03.09 16:00	2 óra	Mali Völgyesi Garai Zavadil	Értekezlet. Döntés: Zavadil elkészíti a 2.1 és 2.2.2 fejezeteket és a fedőlapot. Mali elkészíti a 2.3 fejezetet. Garai elkészíti a 2.5 és 2.6 fejezeteket. Völgyesi feladata a 2.4 fejezet elkészítése. Varga elkészíti a 2.2.1, 2.2.3, 2.2.4, 2.2.5 fejezeteket.
2023.03.10 11:00	2 óra	Mali	2.3 Leírás mezők kitöltése
2023.03.10 15:00	2,5 óra	Zavadil	2.2.2 bekezdés megírása az egyeztetett követelmények alapján, 2.1.1, 2.1.5 megírása.
2023.03.10 17:30	2,5 óra	Varga	Tevékenység: A 2.2.1, 2.2.3, 2.2.4 bekezdések megírása
2023.03.10 14:00	1 óra	Zavadil	Fedőlap megírása, napló kiegészítése
2023.03.11 10:00	2 óra	Völgyesi	2.4 Use-case leírások, use-case diagram
2023.03.11 12:00	1,5 óra	Mali	2.3 többi mező kitöltése
2023.03.11 15:15	2,5 óra	Garai	Projekt terv(2.6) és Szótár(2.5) megírása
2023.03.12 12:00	1 óra	Varga	A 2.2.1 bekezdés átírása mert nem megfelelő a követelményeknek.
2023.03.12 13:00	1 óra	Zavadil	2.1.* bekezdések ellenőrzése, bővítése.

## 3. Analízis modell kidolgozása

### 3.1 Objektum katalógus

#### 3.1.1 Game

Nyilvántartja a játék komponenseit, az aktív játékost és akciójának számát, számolja a két csapat pontjait. Rajta keresztül lehet elindítani a játékot, interfésszt biztosít a grafikus objektum számára: továbbítja az üzeneteket (pl. 'eltöröm' gomb lenyomása) az egyes komponensek (pl. cső) felé. Rajta keresztül lehet mozgatni a csöveket.

#### 3.1.2 Pipe

A játékszabályok szerinti cső funkciót valósítja meg. Nyilvántartja, hogy hányan állnak rajta, lyukas-e, van-e benne víz, kit kivel köt össze és engedélyezi vagy megtiltja játékosoknak hogy rálépjenek. minden kör végén, ha lyukas és van benne víz, a szabotőröknek ad pontot és kifolyik belőle a víz.

#### 3.1.3 Pump

Megvalósítja a játékszabályok szerinti pumpa funkciót. Van víztározója melynek nyilvántartja a vízszintjét, minden kör végén pumpál a kimeneti csövébe a víztározójából, ha abban volt víz. A bemeneti csövén ha van víz és a víztározójában nincs, akkor ugyancsak a kör végén rak a bele vizet. Nyilvántartja, hogy milyen játékosok állnak rajta. minden kör végén van egy kis esélye arra, hogy elromlik.

#### 3.1.4 Cistern

Nyilvántartja, hogy van-e bemeneti csöve és ha igen és van benne víz akkor pontot ad a szerelő játékosoknak a kör végén. Ezen kívül nyilvántartja, hogy van-e nála pumpa, amit a szerelő játékosok felvehetnek. minden kör végén van egy kis esélye arra, hogy generál egy ilyen pumpát, ha még nincs rajta.

#### 3.1.5 Source

Nyilvántartja, hogy van-e kimeneti csöve és ha igen, akkor tesz bele vizet a kör végén.

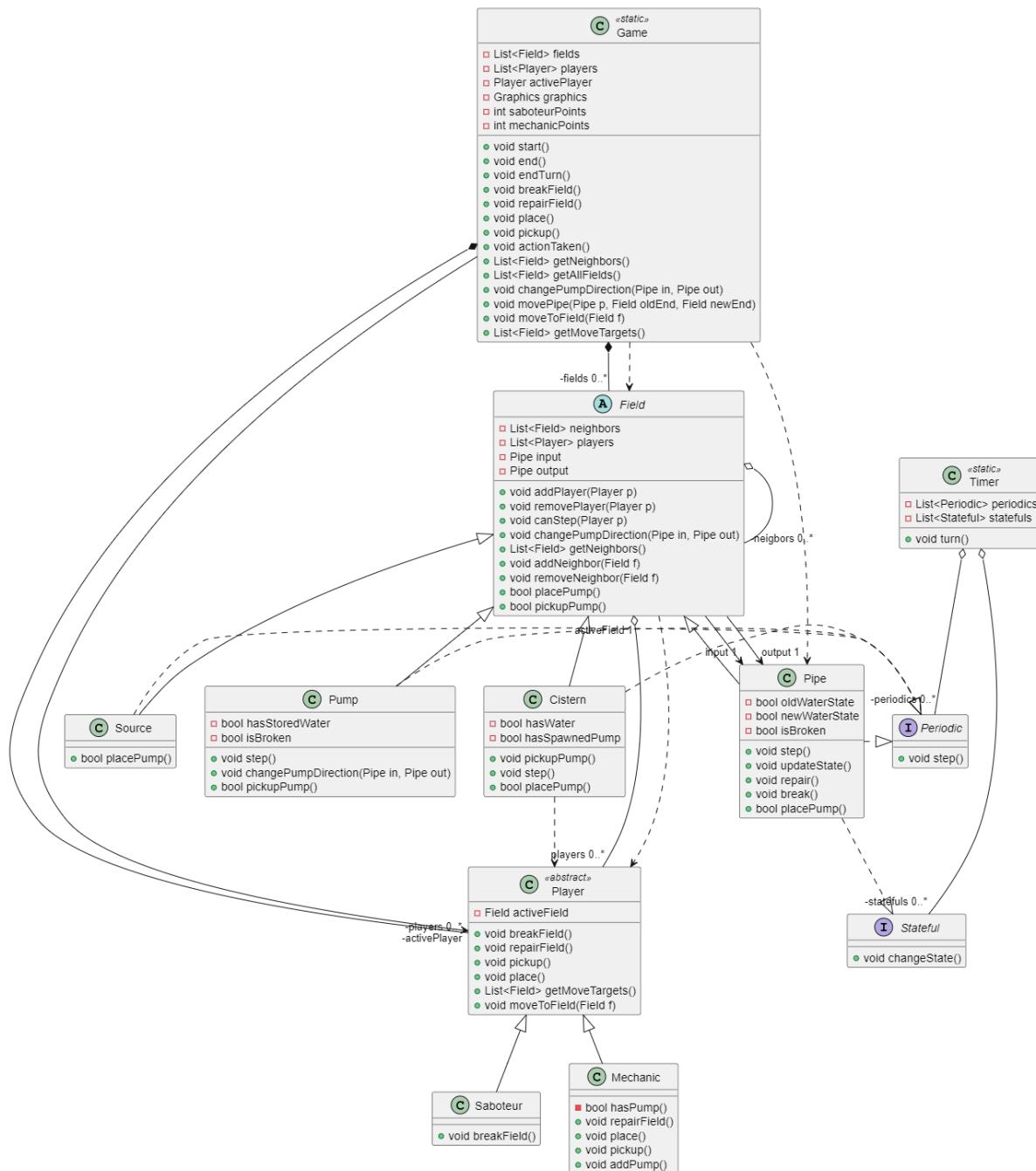
#### 3.1.6 Mechanic

A játékszabályok szerinti szerelő. Áthelyezheti bármely cső egy végét bárhova máshova, átállíthatja azon puma ki és/vagy bemeneti csövét, amin áll, mozoghat a hálózaton, nyilvántartja, hogy jelenleg melyik mezőn áll. Felvehet egy pumpát egy ciszternáról, és leteheti egy cső közepére. Megszerelhet egy elromlott pumpát és megfoltozhat egy lyukas csövet ha rajta áll.

#### 3.1.7 Saboteur

A játékszabályok szerinti szabotőr. Áthelyezheti bármely cső egy végét bárhova máshova, átállíthatja azon puma ki és/vagy bemeneti csövét, amin áll, mozoghat a hálózaton, nyilvántartja, hogy jelenleg melyik mezőn áll. Kilyukaszthat egy csövet amin áll.

## 3.2 Statikus struktúra diagramok



### 3. Analízis modell kidolgozása

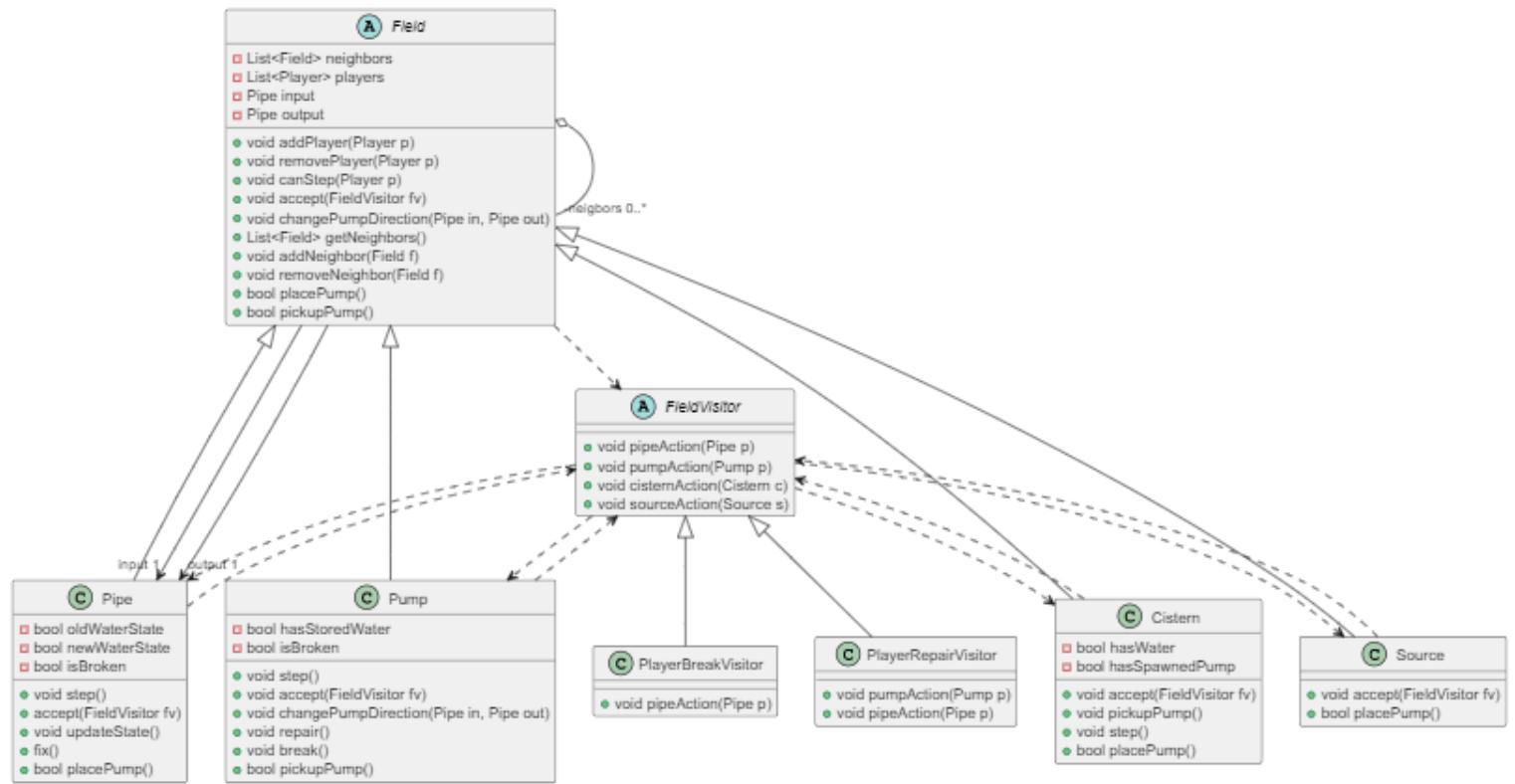
*crazy\_prog\_team*

*Az osztályok attribútumaihoz implicit getter és setter függvények tartoznak, amiket az átláthatóság érdekében nem tüntetünk fel. Azokat a változókat, amelyek egy másik osztály objektumát/objektumait tárolják, az átláthatóság érdekében a vonalak (pl. aggregáció, asszociáció, kompozíció) végein jelöltük a multiplicitásukkal együtt. Emelett az átláthatóság érdekében az osztály leírásában is szerepelnek ezek a változók, azonos névvel.*

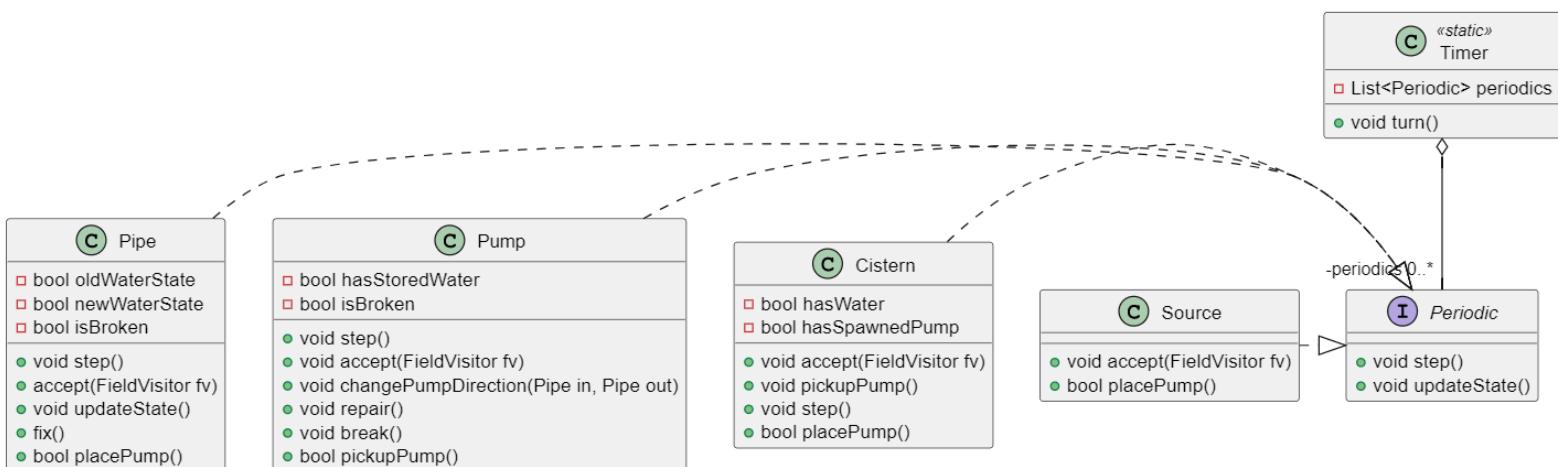
*Az osztálydiagramon az első képen nincsenek ábrázolva a dependenciák (dependency, szaggatott vonal és sima nyíl) a többi kapcsolat átlátásának érdekében. Azonban így az osztálydiagramban szigetek alakultak ki, ezért a későbbi ábrákon már látszanak a dependenciák.*

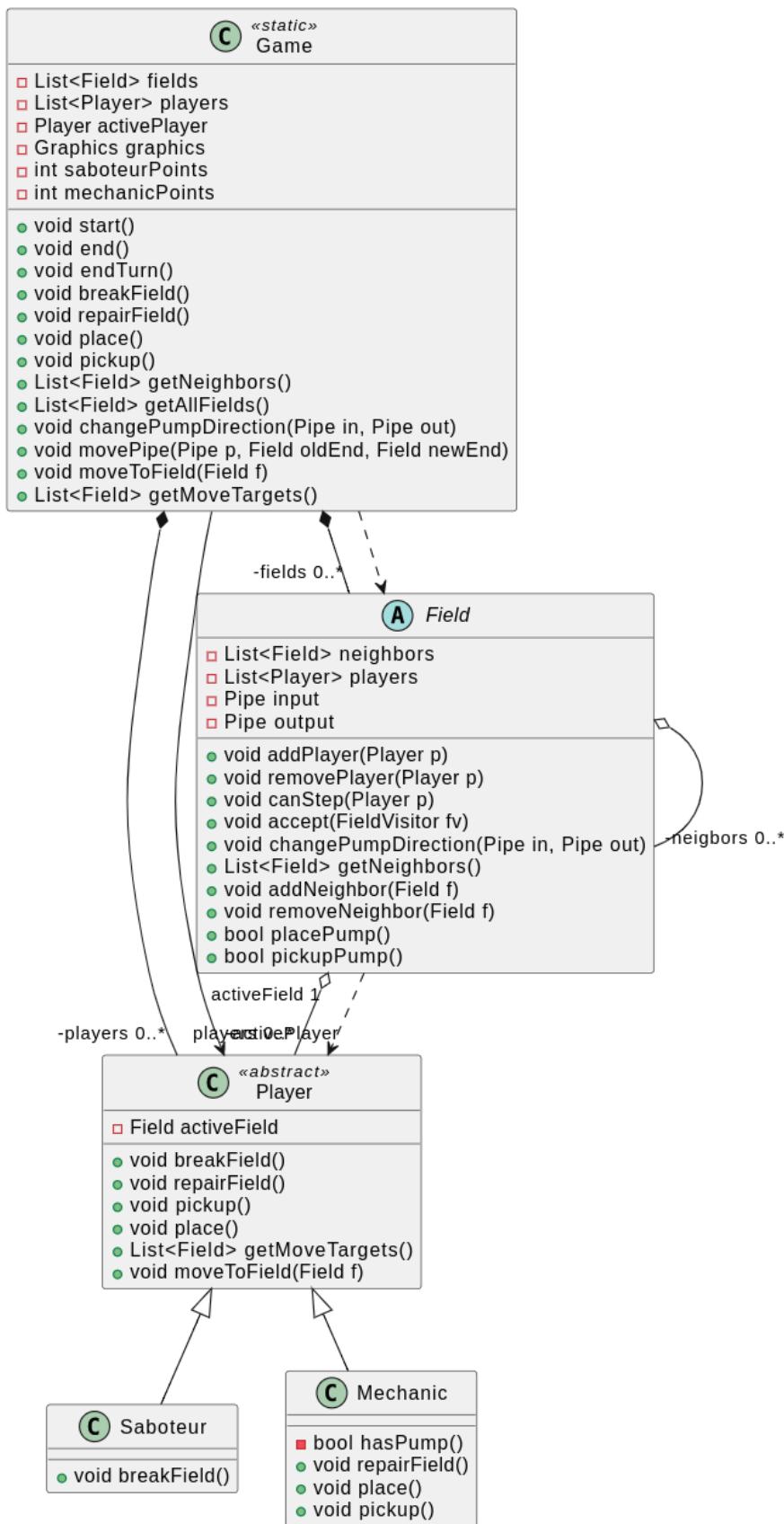
## Osztálydiagram feldarabolva

### Field csoport



### Periodic csoport



**Player csoporthoz**

***Game-Graphics csoport***

### 3.3 Osztályok leírása

#### 3.3.1 Cistern

- **Felelősség**

Felelőssége, hogy eltárolja folyt-e bele víz a körben (mert, ha igen, akkor pontot kapnak a szerelők). Emellett tárolja, hogy van-e rajta még fel nem vett pumpa, melyet a szerelők felvehetnek.

- **Ősosztályok**

Field

- **Interfészek**

Periodic

- **Attribútumok**

- **bool hasWater:** számon tartja ha kap vizet az adott körben.
- **bool hasSpawnedPump:** számon tartja, hogy van-e rajta fel nem vett pumpa

- **Metódusok**

- **void accept(FieldVisitor fv):** Átvesz egy visitor objektumot
- **void takePump(Player p):** Ha van pumpa elkészülve akkor ezt a paraméterül kapott játékosnak odaadja.
- **void step():** minden körben ha van benne víz akkor a Game objectumnak a pontnövelő függvényét meghívja.
- **bool placePump():** Hamis-t ad vissza, mert nem lehet erre a mezőre elhelyezni Pumpát.

#### 3.3.2 Field

- **Felelősség**

Absztrakt osztály, melynek funkciója hogy összegyűjtse azokat az objektumokat, amire a játékosok rá tudnak lépni.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **List<Player> players:** Számoltartja, hogy milyen játékosok állnak rajta
- **neighbours:** Számoltartja a szomszédos Field-eket
- **Pipe input:** A bemeneti cső
- **Pipe output:** A kimeneti cső

- **Metódusok**

- **void addPlayer(Player p):** Felvesz egy új játékosat a players tömbbe. Ez a játékos most már ezen az objektumon van

- **void removePlayer(Player p):** A paraméterül kapott játékost kiveszi a players tömbből, ezzel azt jelezve hogy a játékos lelépett erről a mezőről
- **void canStep(Player p):** Ellenőri hogy ha az adott mezőre rá lehet-e lépni, az adott játékosnak.
- **void acceptFieldVisitor(FieldVisitor v):**
- **List<Field> getNeighbors():** Továbbad egy listát, hogy kik a szomszédos mezők.
- **void addNeighbor(Field f):** a paraméterül kapott mezőt hozzáveszi a szomszédok listájához, tehát egy új objektummal szomszédos a jelenlegi mező
- **void removeNeighbour(Field f):** a paraméterben kapott mezőt eltávolítja a szomszédok listájából, tehát ez a szomszéddal megszünt a kapcsolata

### 3.3.3 FieldVisitor

- **Felelősség**

A mezők által átvett Visitor osztályok öse. Benne vannak definiálva a különböző mező fajták (cső, pumpa, ciszterna, forrás) által meghívandó metódusok (ha cső veszi át az egyik Visitor-t, akkor a pipeAction metódust hívja meg, stb.).

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

Nincsenek

- **Metódusok**

- **void pipeAction(Pipe p):** Az a metódus, amelyet egy cső fog meghívni, ha átvesz egy visitor-t
- **void pumpAction(Pump p):** A pumpák által meghívandó metódus
- **void cisternAction(Cistern c):** A ciszternák által meghívandó metódus
- **void sourceAction(Source s):** A források által meghívandó metódus

### 3.3.4 Game

- **Felelősség**

A Game osztály felelőssége a játék mindenkorai állapotának a tárolása, irányítása. Ebbe bele kell érteni a játékosok listájának jegyzését, a mezők számértartását, az éppen soron lévő játékos számértartását, a csapatok pontjainak számértartását, és a grafikáért felelős Graphics objektum és a körök vezénylését.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **List<Field> fields:** A mezőket számontartó lista
- **List<Player> players:** A karaktereket számontartó lista
- **Player activePlayer:** Az aktív karakter referenciája
- **Graphics graphics:** A grafikáért felelős objektum referenciája
- **int saboteurPoints:** A szabotőr játékosok pontja
- **int mechanicPoints:** A szerelő játékosok pontja

- **Metódusok**

- **void start():** A játékot indító metódus
- **void endTurn():** A kör váltást megkezdő metódus. Kör váltáskor a periodikus pályaelemek lépnek egyet, majd a soron következő játékos lesz aktív
- **void breakField():** Ezt a metódust meghívva a jelenlegi aktív karakter elrontja az éppen aktuális helyzetének megfelelő mezőt (persze, ha olyan mezőt, amit elronthat)
- **void repairField():** Ezt a metódust meghívva a jelenlegi aktív karakter megjavítja az éppen aktuális helyzetének megfelelő mezőt
- **void place():** Ezt a metódust meghívva az aktív karakter lehelyezi a nála lévő pumpát
- **void pickup():** Ezt a metódust meghívva az aktív karakter felvesz egy pumpát
- **List<Field> getNeighbors():** Ez a metódus visszaadja az aktív karakter mezőjének szomszédait
- **List<Field> getAllFields():** Visszaadja a mezők listáját
- **void changePumpDirection(Pipe in, Pipe out):** Beállítja a játékos helyzetének megfelelő pumpa ki- és bemenetét
- **void movePipe(Pipe p, Field oldEnd, Field newEnd):** Átállítja a kiválasztott cső egyik végét
- **void moveToField(Field f):** Az aktuálisan aktív karaktert az átadott mezőre mozgatja
- **List<Field> getMoveTargets():** Visszaadja azon mezőket, amelyekre az aktív karakter léphet
- **void buttonPushed(Button b):**

### 3.3.5 Graphics

- **Felelősség**

Az objektumok megjelenítése a képernyőn.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **Game game:** Ismeri a játék objektumot.

- **Metódusok**

- **void renderMap():** Megjeleníti a képernyőn a játék aktuális állapotának megfelelő képet, benne a mezőkkel, karakterekkel, gombokkal, listákkal stb.

### 3.3.6 Mechanic

- **Felelősség**

A Player osztályhoz képest plusz felelőssége, hogy számolja ki, hogy van-e a karakternél pumpa, és ezt le tudja rakni. Emellett javítani is tud mezőket.

- **Ősosztályok**

Player

- **Interfészek**

Nem valósít meg interface-t.

- **Attribútumok**

- **bool hasPump:** Jelöli hogy ha a szerelőnél van-e pumpa.

- **Metódusok**

- **void place():** Ha van nála pumpa és az activeField-re le lehet tenni ezt, leteszi.
- **void pickup():** A ciszternától felvesz egy pumpát. Ekkor a hasPump értéke megváltozik.
- **void repairField():** Azt a mezőt amin áll megjavítja.

### 3.3.7 Periodic

- **Felelősség**

Lehetővé teszi hogy a hálózat objektumai sorban meghívodjanak egy-egy körben. Ezalatt, minden típusától függően definiál egy viselkedést. Ezen viselkedések összessége alkotja a hálózat folyamát.

- **Metódusok**

- **void step():** minden objektum ami implementálja, ebbé írja le a sajátos viselkedését.
- **void updateState():** A step után, vagy végén hívódik meg. A kör utolsó lépése hogy az adott objektum számítsa ki ha megy rajta át víz, vagy nem.

### 3.3.8 Pipe

- **Felelősség**

Felelőssége, hogy számolja ki, volt-e víz benne az előző körben, és hogy lesz-e benne víz a következő körben. Azt is számolja ki, hogy el van-e törve.

- **Ősosztályok**

Field

- **Interfész**

Periodic

- **Attribútumok**

- **bool oldWaterState:** azt jegyzi le hogy ha van benne víz a legutolsó kör kezdete óta. Ez számít akkor amikor meg kell jeleníteni ezt az elemet.

- **bool newWaterState:** minden körben az updateState() hívásra ennek a változónak az értékét átmásolja az oldWaterState változóba.
- **bool isBroken:** azt irja le ha a pipe jelenlegi állapota el van törve vagy nem.

- **Metódusok**

- **void step():** Ha az állapota el van törve, akkor az oldWaterState és a newWaterState egyaránt üres, illetve a szabotörök kapnak pontot. Különben ellenőrzi hogy hogy viszonyul egymáshoz az oldWaterState és a newWaterState,
- **void accept(FieldVisitor fv):** Interakcióba lép egy FieldVisitor objektummal.
- **void updateState():** Ha különbözik a két állapot (newWaterState) és (oldWaterState), akkor az oldWaterState megkapja a newWaterState értékét.
- **void fix():** A szerelő hívhatja meg ezt a függvényt. Ekkor az isBroken értéke hamis lesz.
- **bool placePump():** Igaz értéket ad vissza, mert a csőre rá lehet tenni egy Pumpát.

### 3.3.9 Player

- **Felelősség**

Absztrakt osztály, mely definiálja a Saboteur és a Mechanic közös viselkedését.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **Field activeField:** Azt jelzi hogy melyik mezőn áll abban a pillanatban.

- **Metódusok**

- **List<Field> getMoveTargets():** Visszaad listát azokról a mezőkről, amire a játékos szabad ebben a körben rálépjen. Felhasználja az alatta lévő mezőnek a getNeighbors() függvényét, és sorban felhasználja a canStep() függvényt, hogy ellenőrizze, hogy ha legális neki átlépnie arra a mezőre.
- **void moveToField(Field f):** A paraméterként kapott mezőre átlép. Ekkor az activeField értéke a paraméterként kapott f lesz. Tudjuk hogy ez csak legális Field-ek közül lehetett.

### 3.3.10 PlayerBreakVisitor

- **Felelősség**

Akkor kerül létrehozásra egy ilyen visitor, ha egy játékos akar elrontani egy mezőt. Csak a pipeAction metódus rontja el a csövet, ami átveszi a visitort, a többi metódus üresen van hagyva (csak a csövet tudják elrontani játékosok).

- **Ősosztályok**

FieldVisitor

- **Interfészek**

Nincs

- **Attribútumok**

Nincsenek

- **Metódusok**

- **void pipeAction(Pipe p):** Az átvett csövön meghívja a break() metódust, ezzel elrontva azt.
- **void pumpAction(Pump p):** Üres
- **void cisternAction(Cistern c):** Üres
- **void sourceAction(Source s):** Üres

### 3.3.11 PlayerRepairVisitor

- **Felelősség**

Akkor kerül létrehozásra egy ilyen visitor, ha egy játékos meg akar javítani egy mezőt. Csak a pipeAction és a pumpAction metódusok nincsenek üresen hagyva, mivel csak csövet és pumpát lehet megjavítani.

- **Ősosztályok**

FieldVisitor

- **Interfészek**

Nincs

- **Attribútumok**

Nincsenek

- **Metódusok**

- **void pipeAction(Pipe p):** Az átvett csövön meghívja a fix() metódust, ezzel megjavítva
- **void pumpAction(Pump p):** Az átvett pumpán meghívja a fix() metódust, ezzel megjavítva
- **void cisternAction(Cistern c):** Üres
- **void sourceAction(Source s):** Üres

### 3.3.12 Pump

- **Felelősség**

A felelősége hogy csomópontot helyezzen a hálózatban, ezzel taktikai előnyt szerezve, akár a szabotőröknek, akár a szerelőknek. Ha a pumpa bemenetén megszűnik az áramlás, akkor még 3 körön keresztül a kimenetet ellátja vízzel, hála egy beépített víztartálynak. Ahogy újra víz megy rajta keresztül, 3 kör alatt visszatölödik a víztartály.

- **Ősosztályok**

Field

- **Interfészek**

Megvalósítja a Periodic interface-t, aminek következtében minden körben elvégzi a step() függvény által definiált viselkedést.

- **Attribútumok**

- **int hasStoredWater:** számon tartja ha van-e a víztartályban víz.
- **bool isBroken:** számon tartja hogy ha a pumpa elromlott-e.

- **Metódusok**

- **void step():** Meghívja a break() metódust. Ha elromlik akkor a kimenetén a víz már nem folyik. Ha nem romlott el, a következő viselkedés definiált: Ha a kimeneti csövén nincs víz és a tartályában van, a kimeneti csőre tesz vizet (ilyenkor nem tűnik el a tartályból a víz). Majd a bemeneti csövéről megpróbál beszívni vizet a tartályába, ha a tartály üres. Amennyiben a bemeneti csőben nincs víz, a tartály üres lesz.
- **void accept(FieldVisitor fv):** Interakcióba lép egy FieldVisitor objektummal.
- **void changePumpDirection(Pipe in, Pipe out):** Beállítja a bemeneti csövet az in paraméter által kapott csőre, a kimeneti csövet az out paraméter által kapott csőre (A bemeneti és kimeneti csövek az **input** és **output** tagváltozók, melyeket a **Pump** a **Field** osztályból örököl).
- **void repair():** Egy Mechanic játékos hívja meg ezt a metódust. Ha az állapota romlott volt akkor, ezentúl már nem romlott.
- **void break():** A step() metódus közben hívódik meg. Random generátor segítségével eldönti hogy ha abban a körben romoljon el vagy sem. Ha elromlik akkor megváltoztatja az állapotát romlottra.
- **bool placePump():** Hamis értékkel tér vissza, mert pumpára nem lehet másik pumpát letenni.

### 3.3.13 Saboteur

- **Felelősség**

Az objektum felelőssége hogy a csöveket rongáljon és mászkáljon a hálózaton, illetve elállja az útját a szerelőknek, akik meg akarják javítani ezt.

- **Ősosztályok**

Player

- **Interfészek**

Nincs interface

- **Attribútumok**

- **Field activeField:** Az a Field, amin jelenleg áll a játékos.

- **Metódusok**

- **void breakField():** Megpróbálja eltörni azt a Field-et, amin áll (activeField). Ehhez segítségül hívja a PlayerBreakVisitor osztályt (Saboteur Breaks Pipe szekvenciadiagram)

### 3.3.14 Source

- **Felelősség**

Végtelen vízforrásnak számít. A kimenetén mindenkor folyik a víz. Ebből lehet eljuttatni vizet a ciszternába.

- **Ősosztályok**

Field

- **Interfészek**

Megvalósítja a Periodic interface-t, aminek következtében mindenkorben elvégzi a step() függvény által definiált viselkedést.

- **Attribútumok**

Nincs attribútum-ja.

- **Metódusok**

- **void accept(FieldVisitor fv):** Interakcióba lép egy FieldVisitor objektummal.
- **bool placePump():** Hamisat ad vissza, mert nem lehet letenni rá pumpát.

### 3.3.12 Timer

- **Felelősség**

A játék köreinek léptetéséért felelős. Ismerve az összes olyan objektumot amit léptetni kell, sorban ezeken meghívja a step() metódust. Ez előtt azonban meghívja az updateState() metódust is rajtuk.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

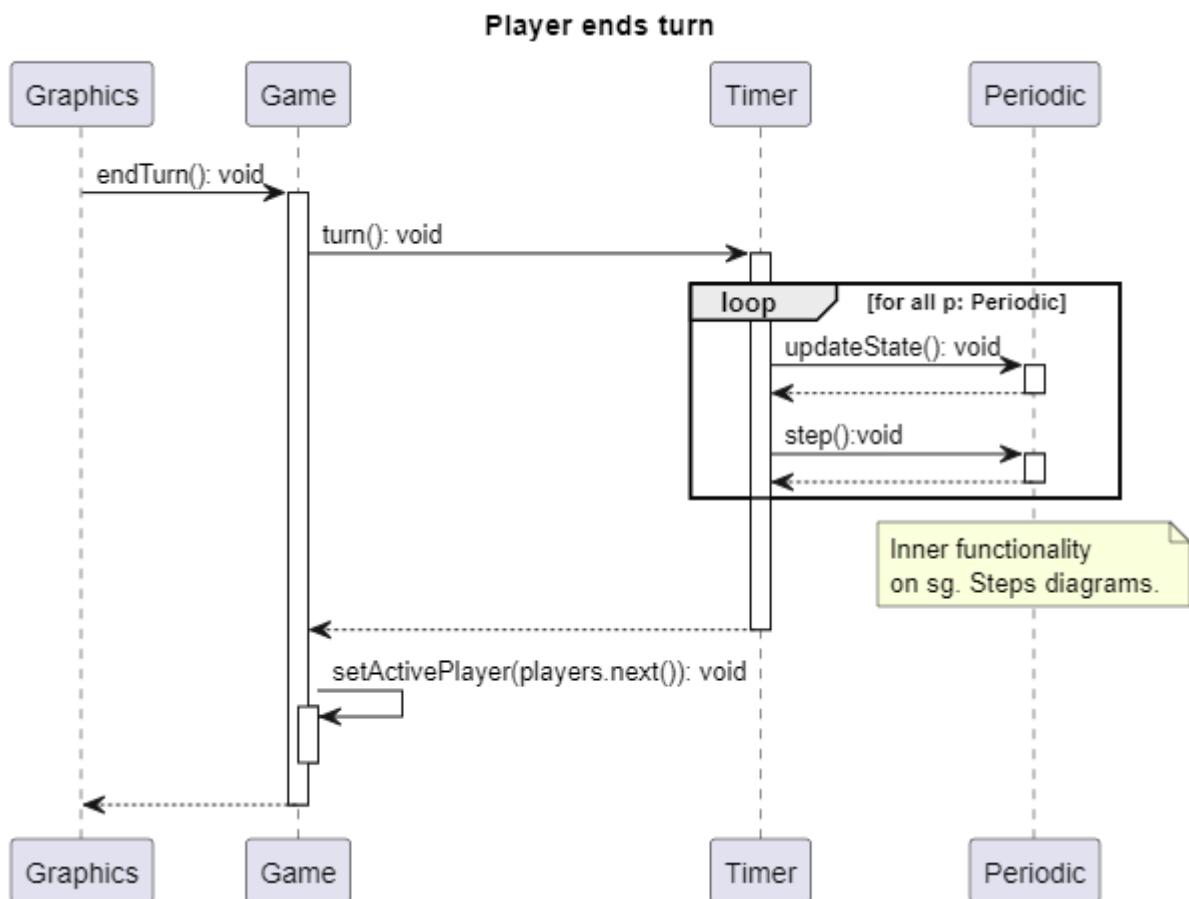
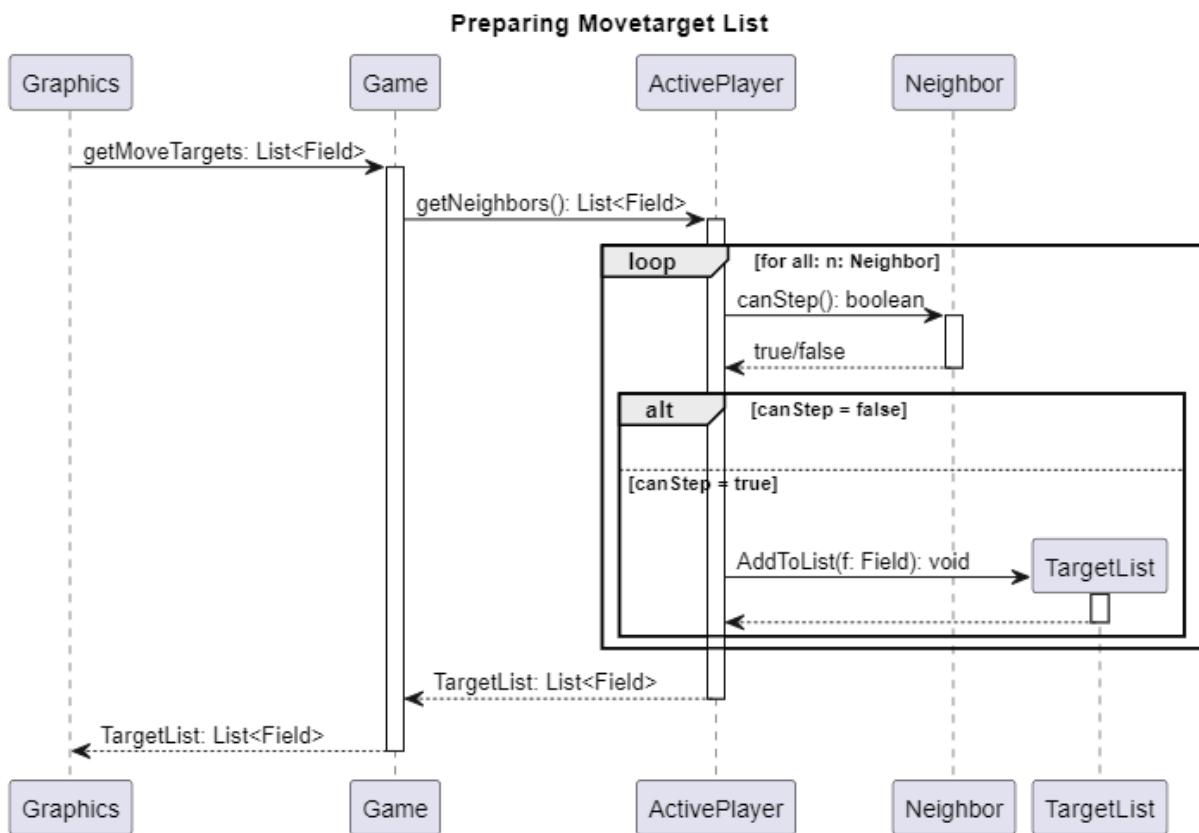
- **List<Periodic> periodics:** Számon tartja az összes olyan objektumot ami megvalósítja a Periodic interface-t.

- **Metódusok**

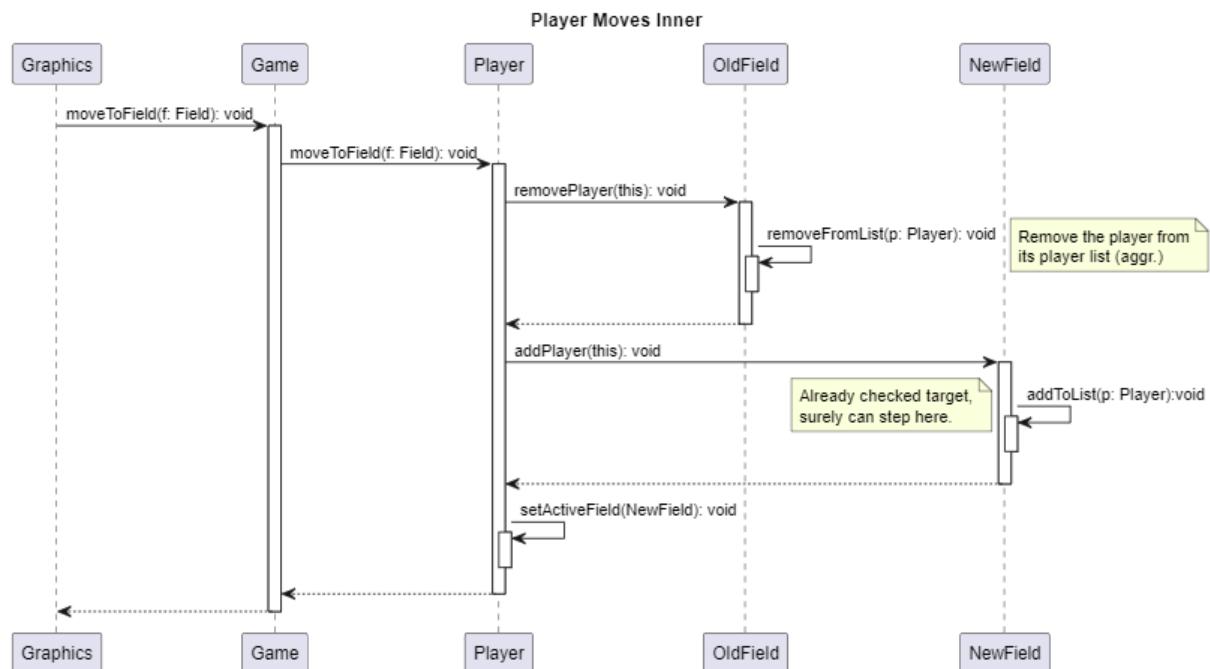
- **void turn():** Egy ciklussal végigmegy a periodics listában lévő objektumokon és mindenkorban meghívja először az updateState() majd a step() függvényt.

## 3.4 Szekvencia diagramok

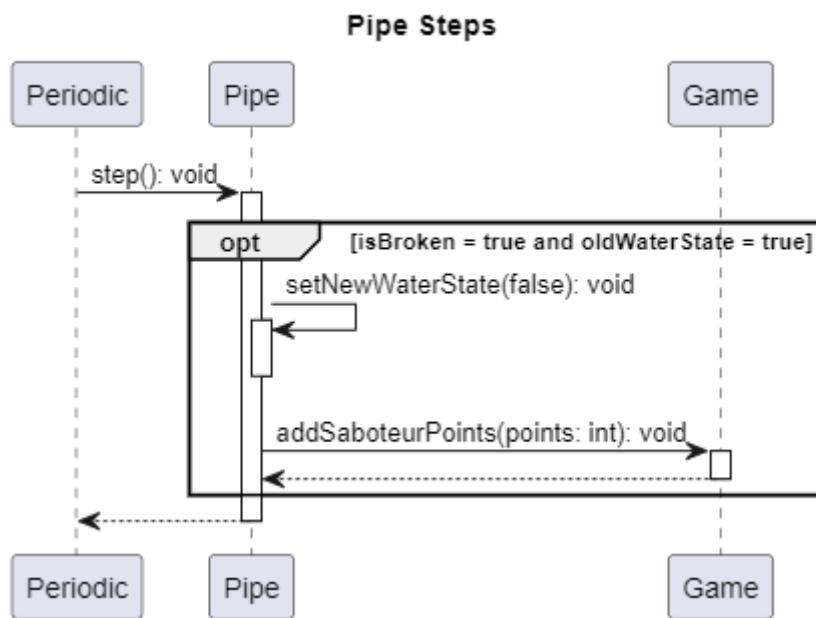
**Player Ends Turn:**

**Preparing MoveTarget List:**

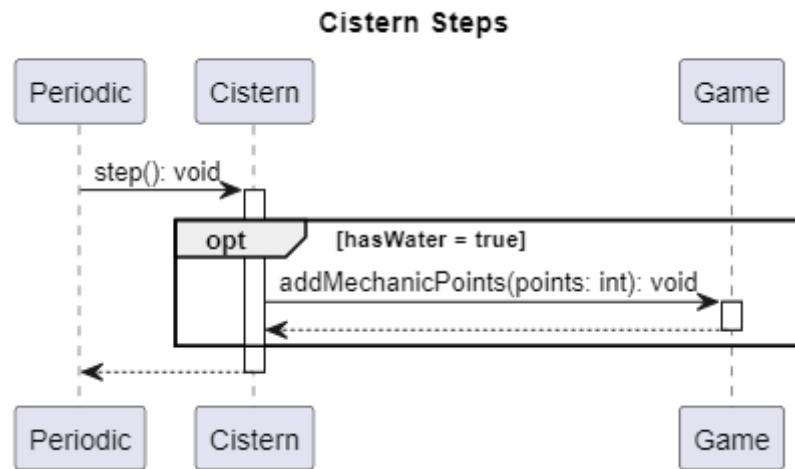
### Player Moves Inner:



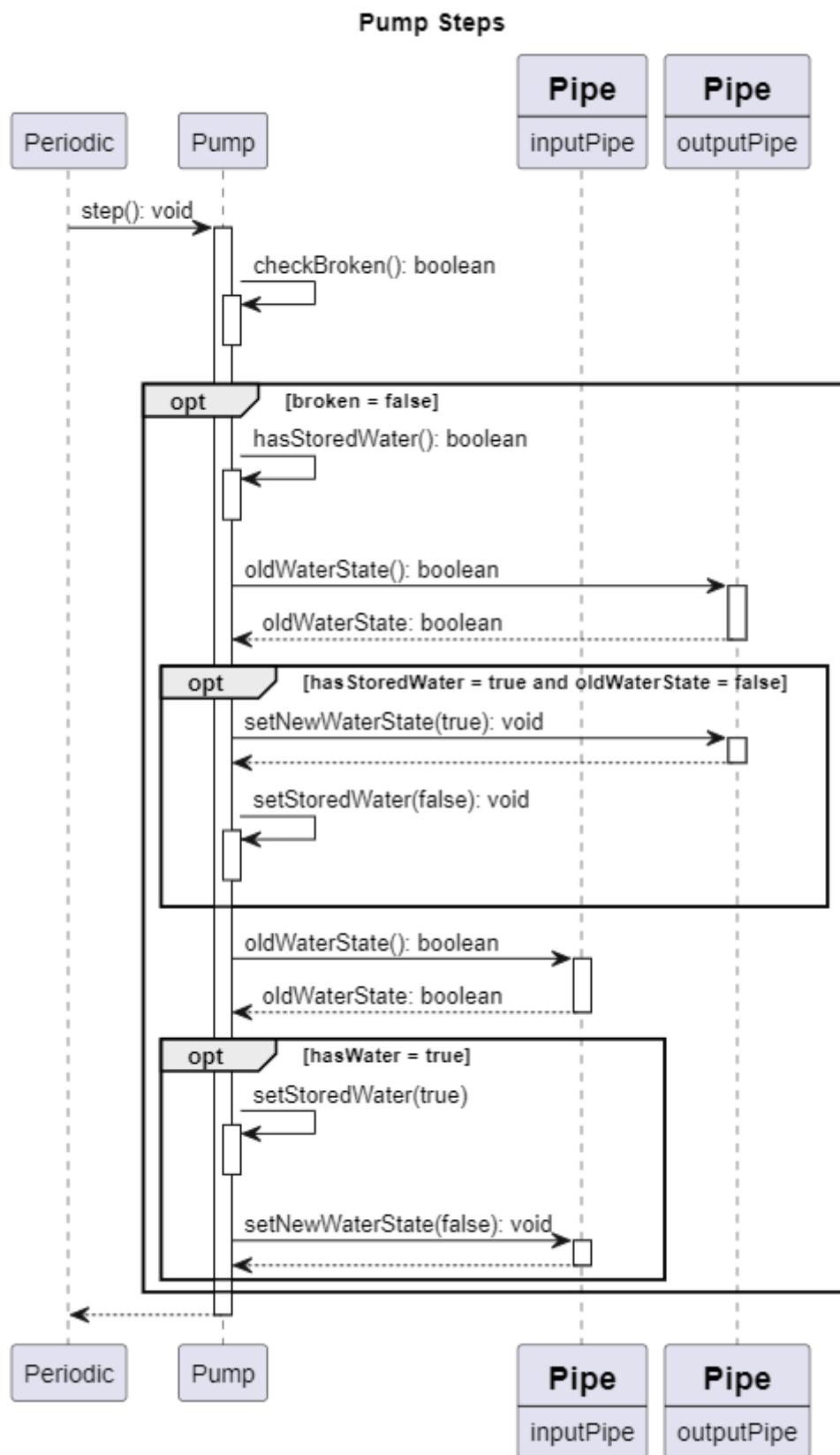
### Pipe Steps:

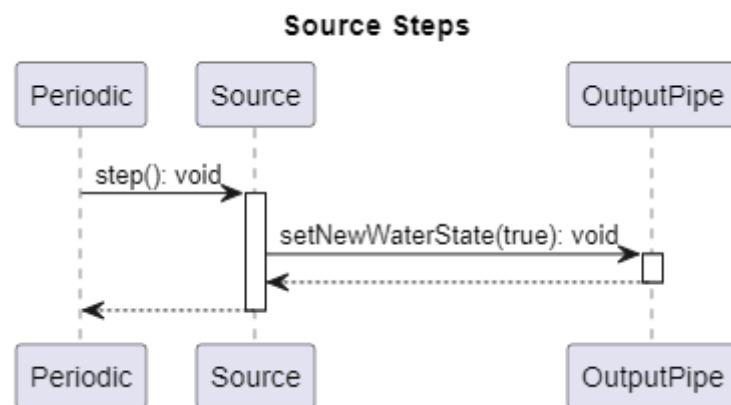
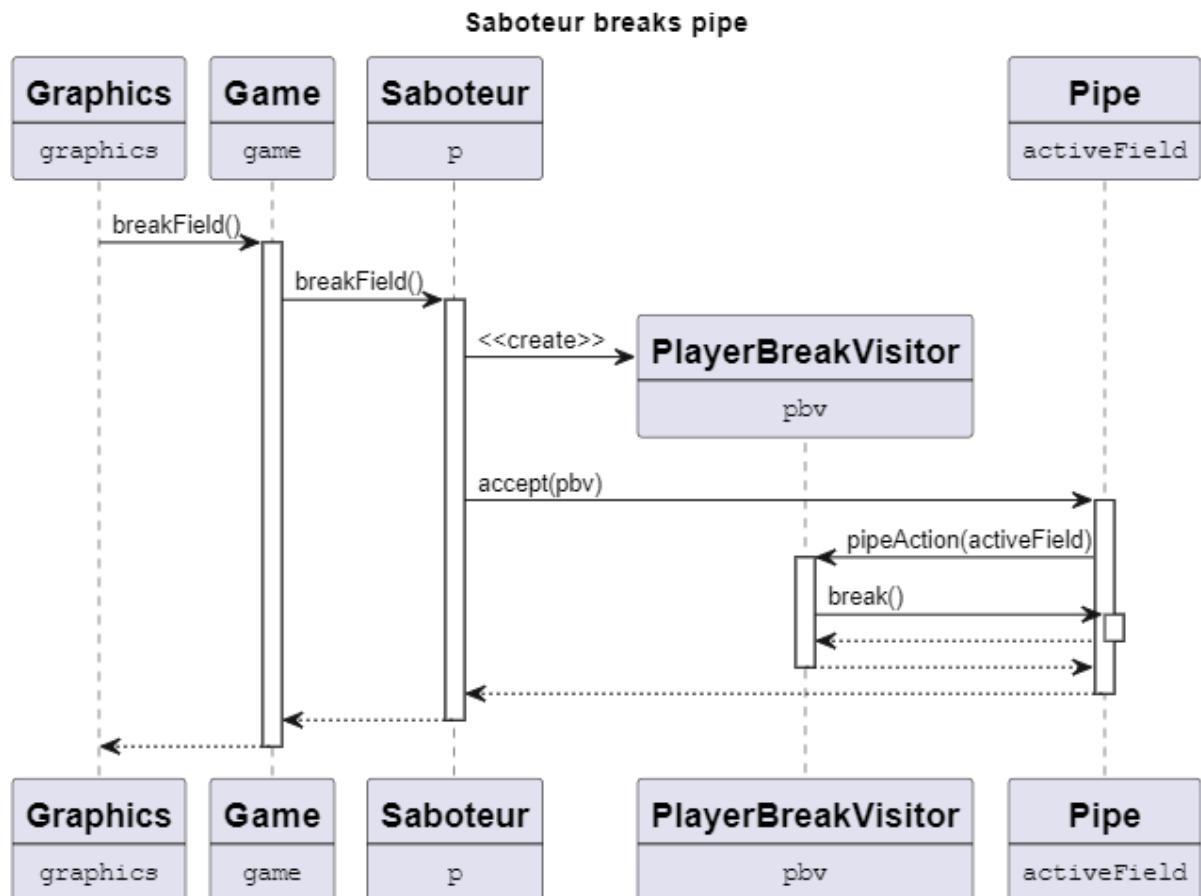


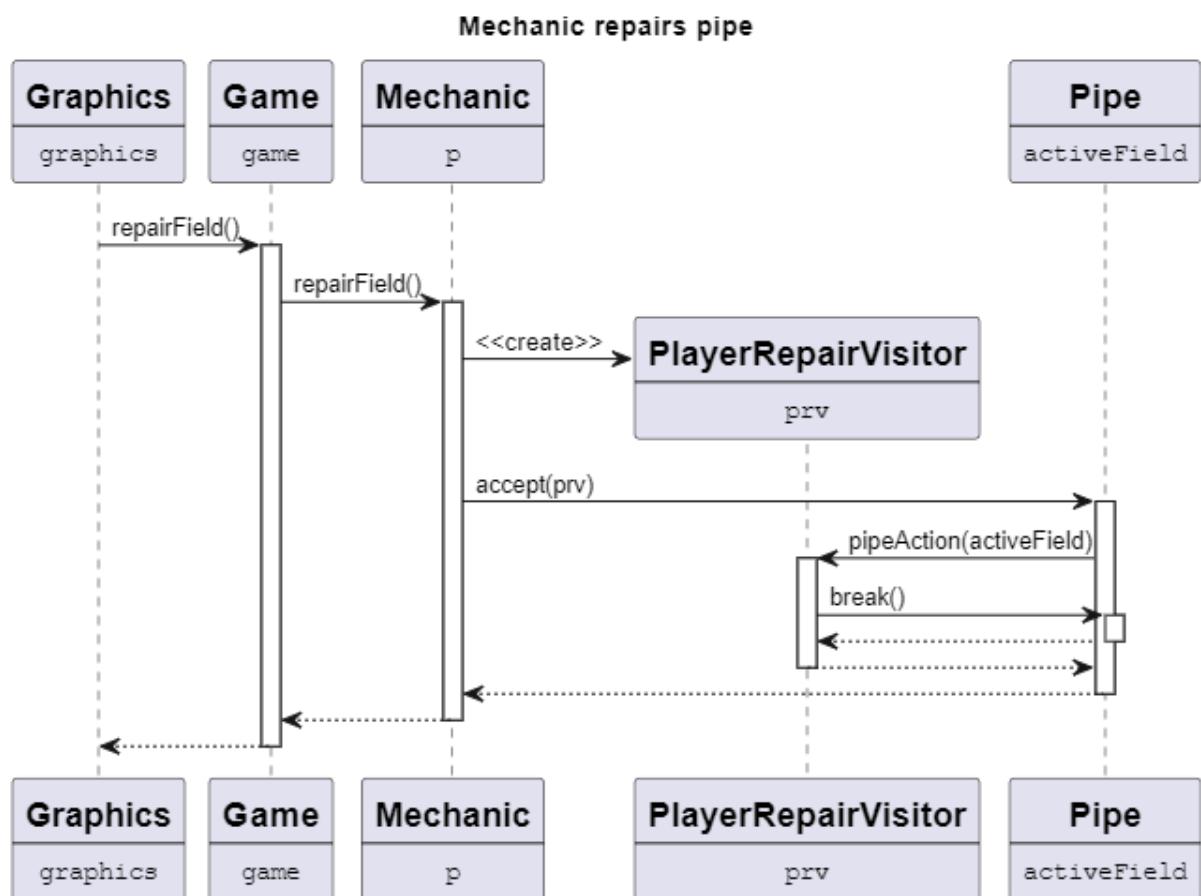
### Cistern Steps:



**Pump Steps:**

**Source Steps:**

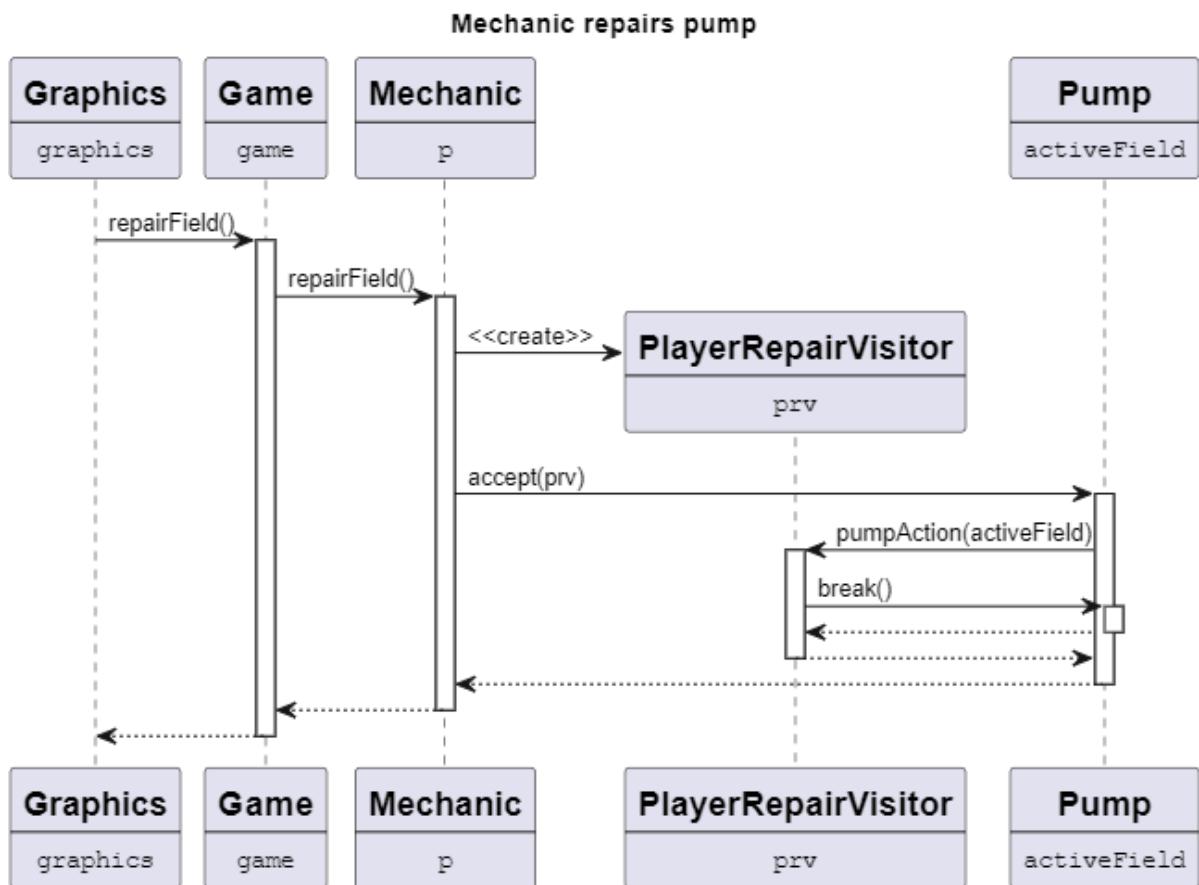
**Saboteur Breaks Pipe:****Mechanic Repairs Pipe:**



**Mechanic Repairs Pump:**

### 3. Analízis modell kidolgozása

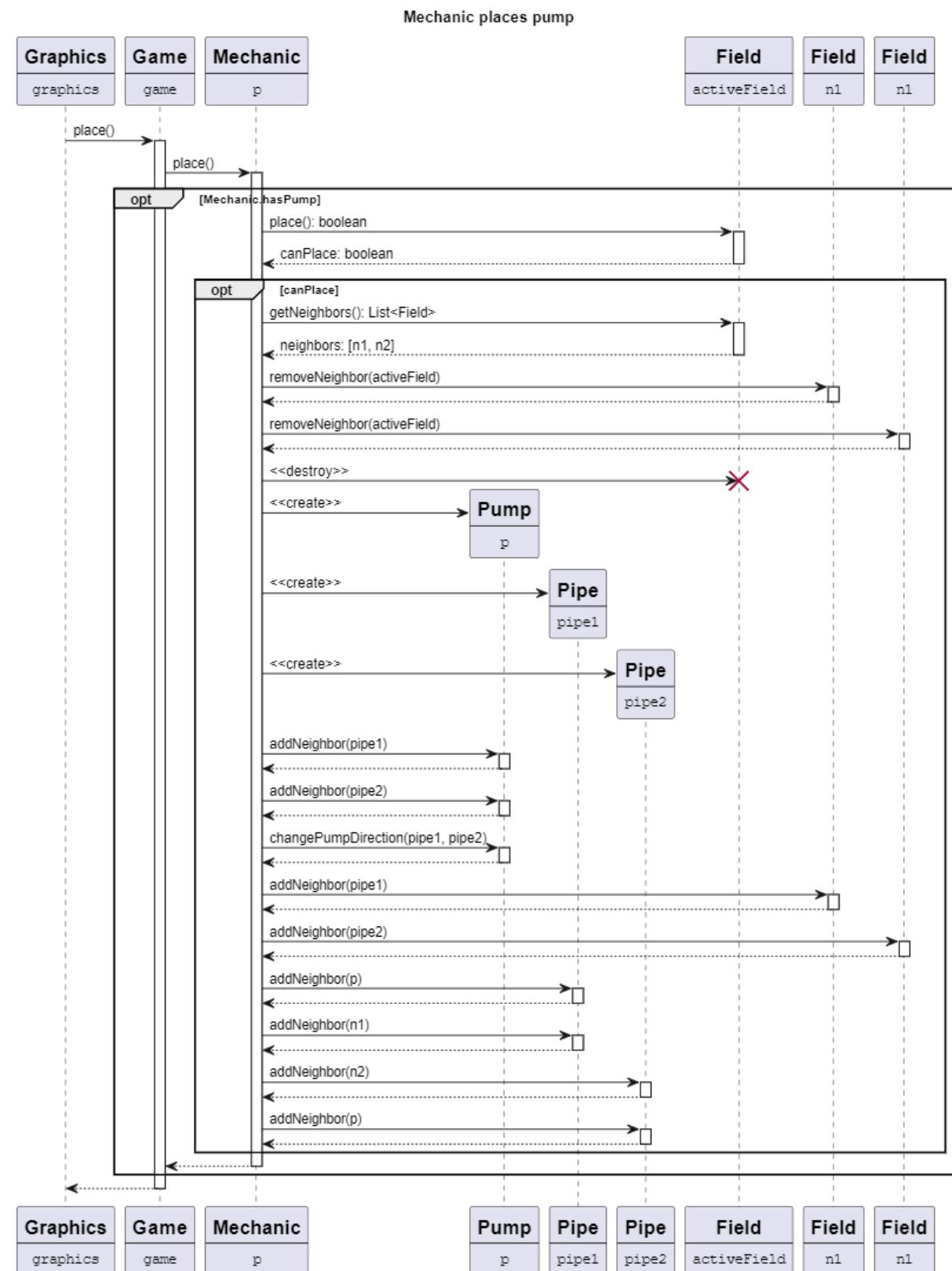
crazy\_prog\_team

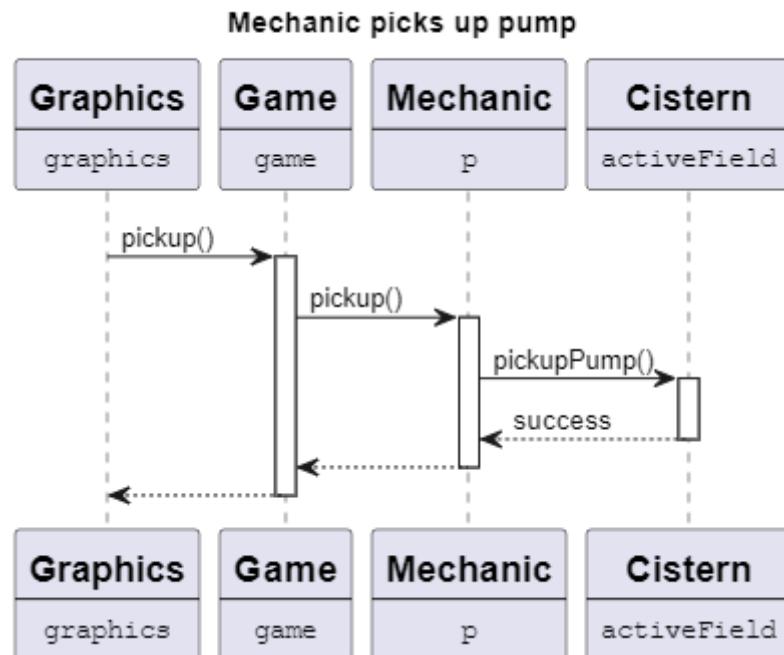


## Mechanic Places Pump:

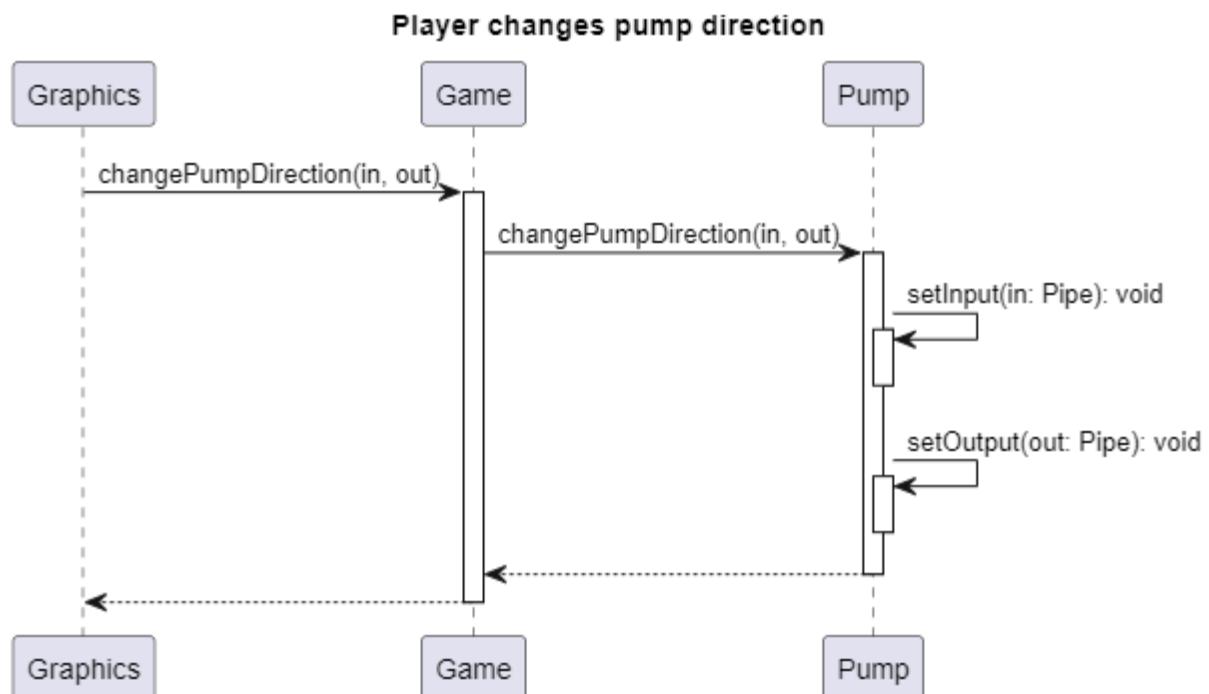
### 3. Analízis modell kidolgozása

*crazy\_prog\_team*

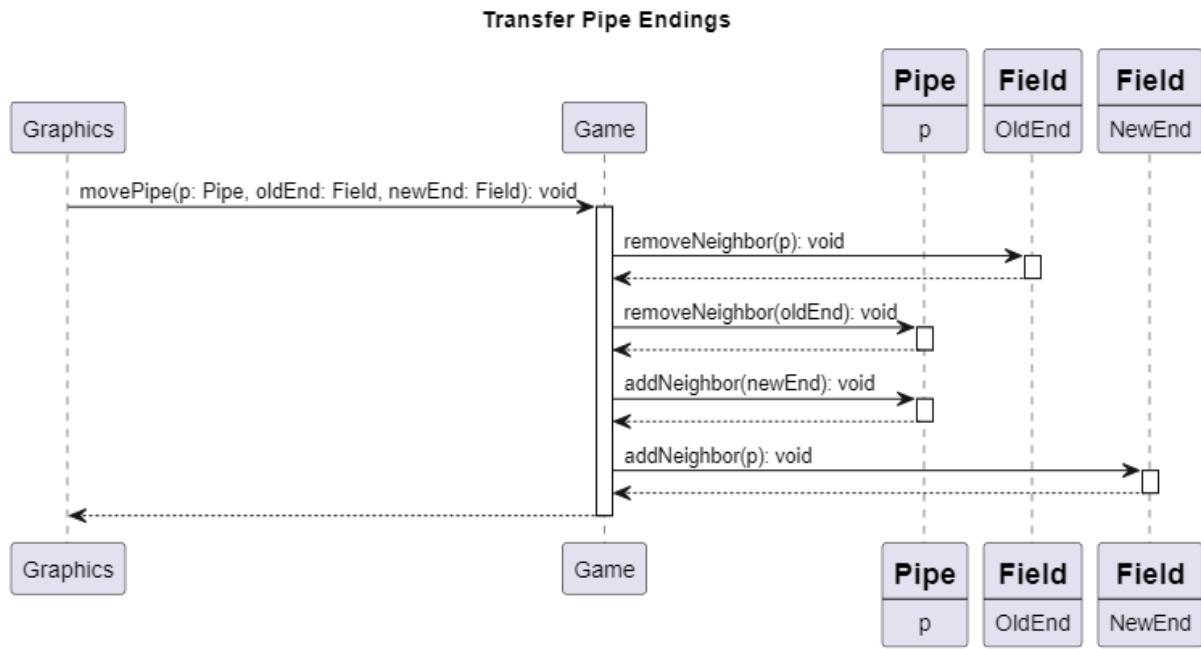




**Player Changes Pump Direction:**

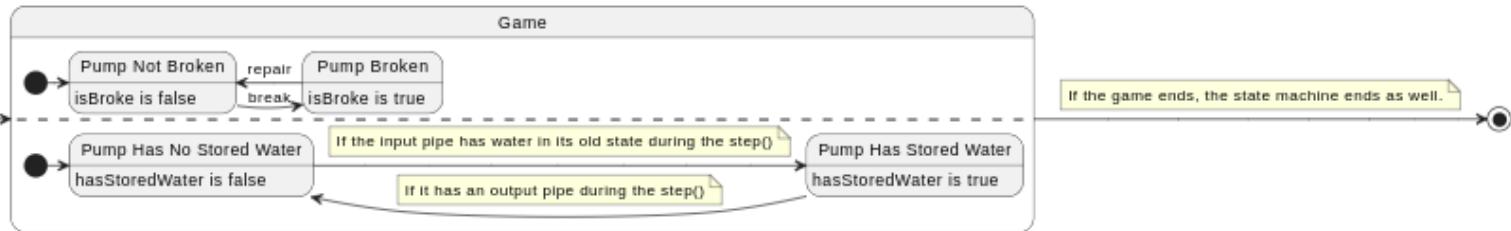


**Transfer Pipe Endings:**



### 3.5 State-chartok

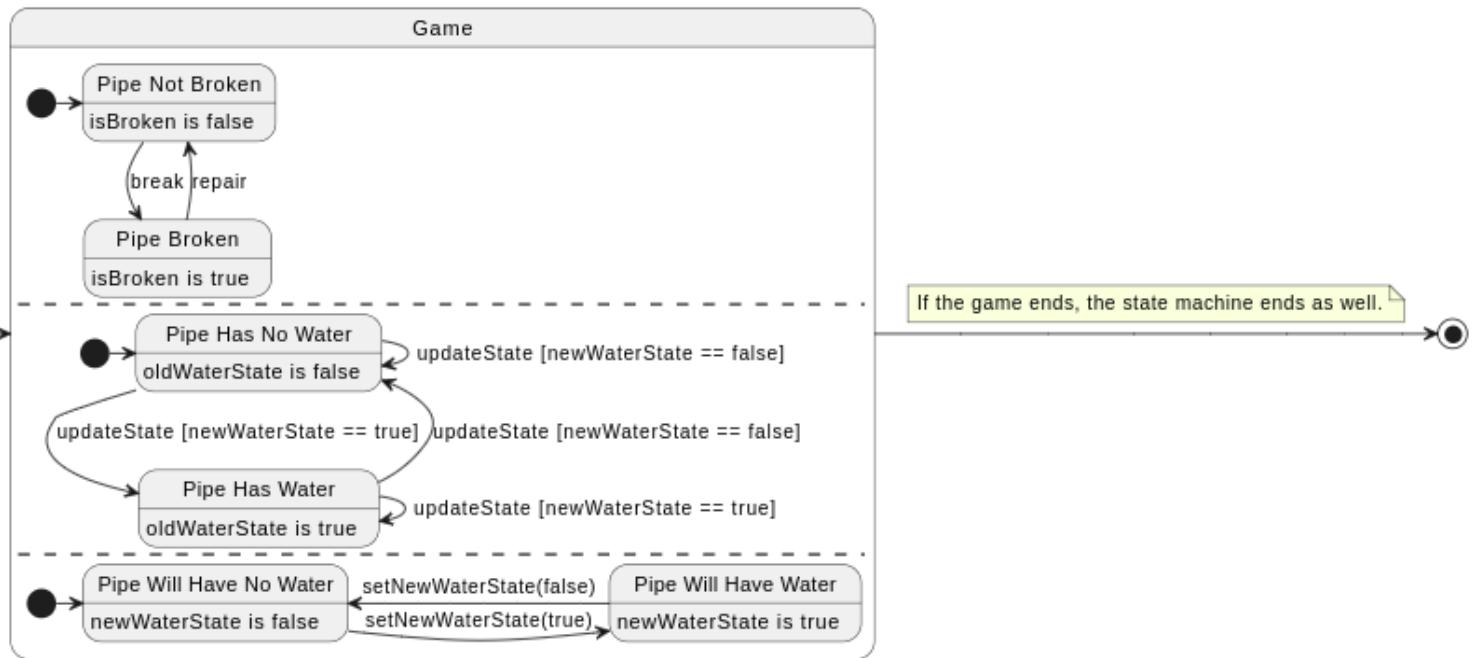
Pump State-chart:



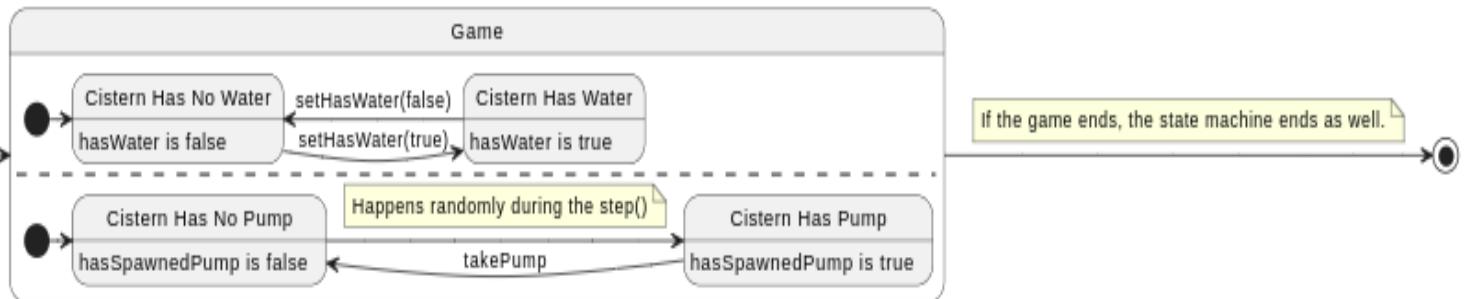
Pipe State-chart:

### 3. Analízis modell kidolgozása

*crazy\_prog\_team*



### Cistern State-chart:



### 3. Analízis modell kidolgozása

*crazy\_prog\_team*

## Napló

Kezdet	Időtartam	Résztvevők	Leírás
2023.03.17. 14:00	3 óra	Garai Mali Varga Völgyesi Zavadil	Értekezlet, közös munka. A program leírása alapján az osztályok és azok felelősségeinek fixálása. Az osztálydiagram struktúrájának megalkotása. Egyes osztályok legfőbb attribútumainak és metódusainak megbeszélése.
2023.03.17. 17:00	2.5 óra	Zavadil	Osztálydiagram alapjainak elkészítése az értekezleten elhangzottak alapján.
2023.03.19. 14:00	5.5 óra	Zavadil	Osztálydiagram bővítése a szekvencia-diagramokkal párhuzamosan. Ezek rögzítése a dokumentumban.
2023.03.19. 14:00	6 óra	Völgyesi	Szekvencia diagramok készítése, felvitele a dokumentumba
2023.03.19. 17:00	3 óra	Varga	3.3 bekezdés kidolgozása
2023.03.19. 15:00	3 óra	Garai	Az állapotgépek elkészítése, felvitele a dokumentumba. (3.5)
2023.03.19. 20:30	2 óra	Garai	Osztálydiagramban hibakeresés és javítás. Osztálydiagram feldarabolása.

### 3. Analízis modell kidolgozása

#### 3.1 Objektum katalógus

##### 3.1.1 Game

*Nyilvántartja a játék komponenseit, az aktív játékost és akciójának számát, számolja a két csapat pontjait. Rajta keresztül lehet elindítani a játékot, interfésszt biztosít a grafikus objektum számára: továbbítja az üzeneteket (pl. ‘eltöröm’ gomb lenyomása) az egyes komponensek (pl. cső) felé. Rajta keresztül lehet mozgatni a csöveket.*

##### 3.1.2 Pipe

*A játékszabályok szerinti cső funkciót valósítja meg. Nyilvántartja, hogy hányan állnak rajta, lyukas-e, van-e benne víz, kit kivel köt össze és engedélyezi vagy megtiltja játékosoknak hogy rálépjenek. minden kör végén, ha lyukas és van benne víz, a szabotőröknek ad pontot és kifolyik belőle a víz.*

##### 3.1.3 Pump

*Megvalósítja a játékszabályok szerinti pumpa funkciót. Van víztározója melynek nyilvántartja a vízszintjét, minden kör végén pumpál a kimeneti csövébe a víztározójából, ha abban volt víz. A bemeneti csövén ha van víz és a víztározójában nincs, akkor ugyancsak a kör végén rak a bele vizet. Nyilvántartja, hogy milyen játékosok állnak rajta. minden kör végén van egy kis esélye arra, hogy elromlik.*

##### 3.1.4 Cistern

*Nyilvántartja, hogy van-e bemeneti csöve és ha igen és van benne víz akkor pontot ad a szerelő játékosoknak a kör végén. Ezen kívül nyilvántartja, hogy van-e nála pumpa, amit a szerelő játékosok felvehetnek. minden kör végén van egy kis esélye arra, hogy generál egy ilyen pumpát, ha még nincs rajta.*

##### 3.1.5 Source

*Nyilvántartja, hogy van-e kimeneti csöve és ha igen, akkor tesz bele vizet a kör végén.*

##### 3.1.6 Mechanic

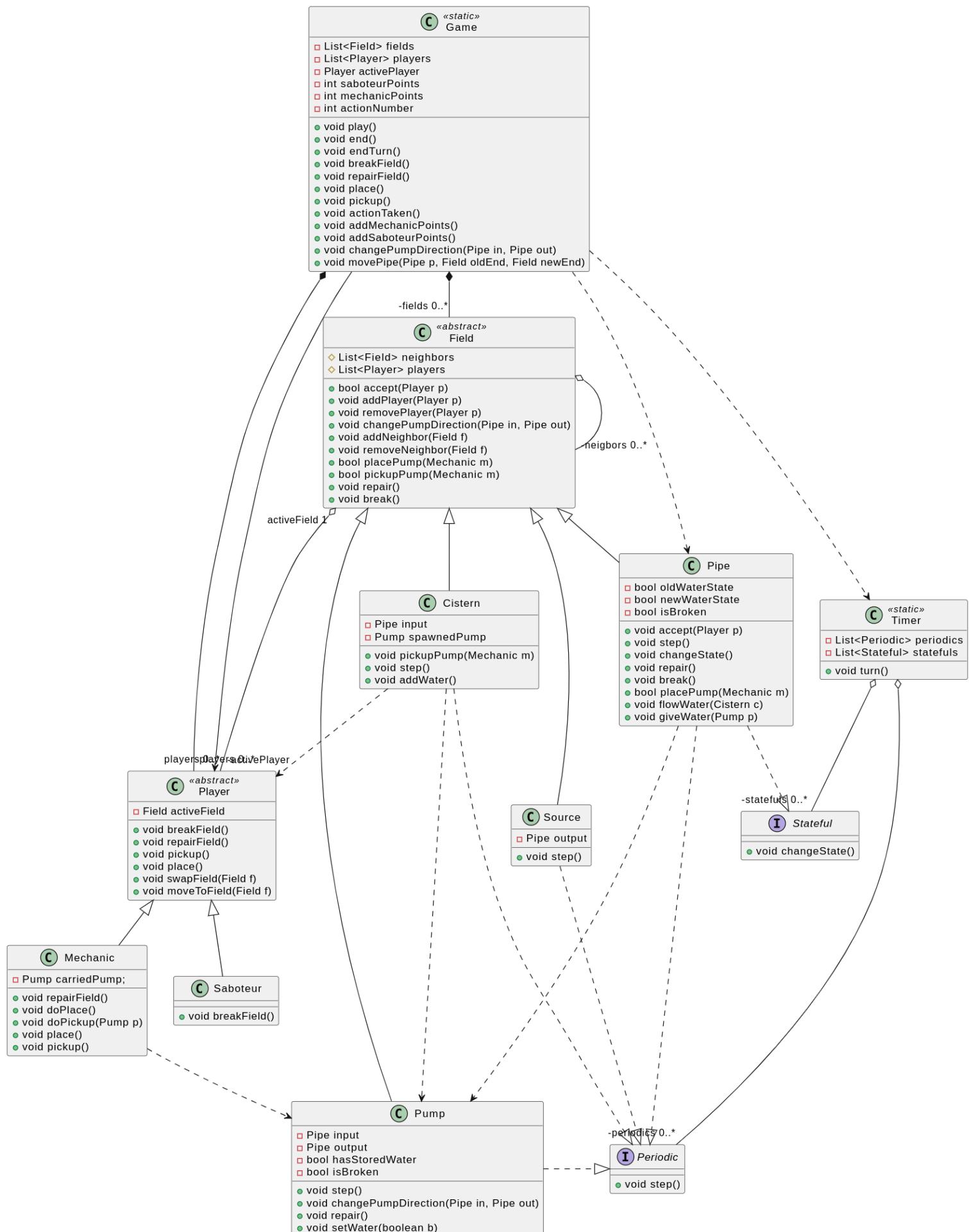
*A játékszabályok szerinti szerelő. Áthelyezheti bármely cső egy végét bárhova máshova, átállíthatja azon puma ki és/vagy bemeneti csövét, amin áll, mozoghat a hálózaton, nyilvántartja, hogy jelenleg melyik mezőn áll. Felvehet egy pumpát egy ciszternáról, és leteheti egy cső közepére. Megszerelhet egy elromlott pumpát és megfoltozhat egy lyukas csövet ha rajta áll.*

##### 3.1.7 Saboteur

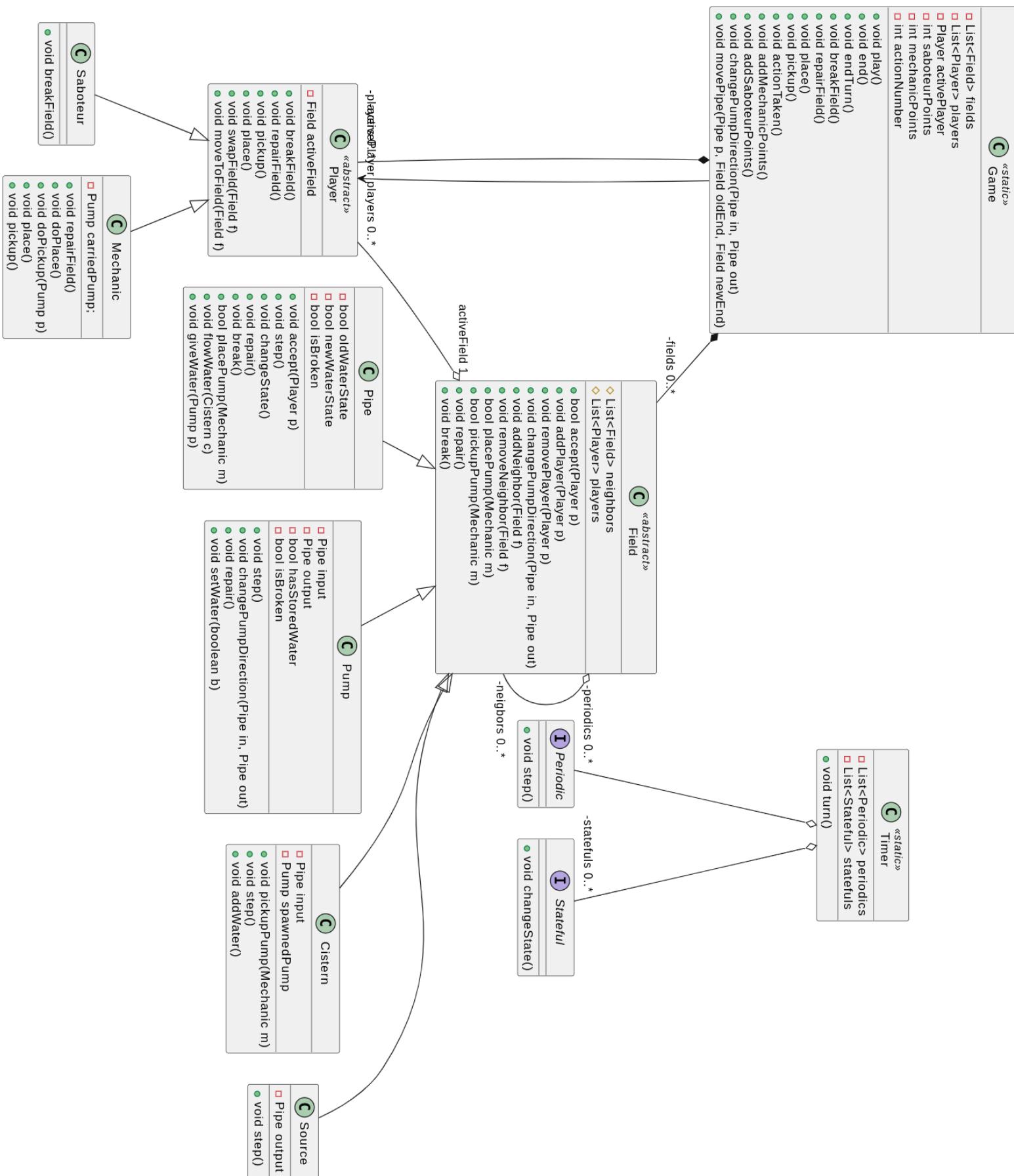
*A játékszabályok szerinti szabotőr. Áthelyezheti bármely cső egy végét bárhova máshova, átállíthatja azon puma ki és/vagy bemeneti csövét, amin áll, mozoghat a hálózaton, nyilvántartja, hogy jelenleg melyik mezőn áll. Kilyukaszthat egy csövet amin áll.*

## **3.2 Statikus struktúra diagramok**

### **3.2.1 Osztálydiagram dependenciákkal együtt**



### 3.2.2 Osztálydiagram dependenciák nélkül



*Az osztályok attribútumaihoz implicit getter és setter függvények tartoznak, amiket az átláthatóság érdekében nem tüntetünk fel. Azokat a változókat, amelyek egy másik osztály objektumát/objektumait tárolják, az átláthatóság érdekében a vonalak (pl. aggregáció, asszociáció, kompozíció) végein jelöltük a multiplicitásukkal együtt. Emelett az átláthatóság érdekében az osztály leírásában is szerepelnek ezek a változók, azonos névvel.*

*Az osztálydiagramon a második képen nincsenek ábrázolva a dependenciák (dependency, szaggatott vonal és sima nyíl) a többi kapcsolat átlátásának érdekében.*

## Osztályok leírása

### 3.2.1 Cistern

- **Felelősség**

Felelőssége, hogy eltárolja folyt-e bele víz a körben (mert, ha igen, akkor pontot kapnak a szerelők). Emellett tárolja, hogy van-e rajta még fel nem vett pumpa, melyet a szerelők felvehetnek.

- **Ősosztályok**

Field

- **Interfészek**

Periodic

- **Attribútumok**

- **Pipe input:** A bemeneti csöve
- **Pump spawnedPump:** Ha van rajta pumpa, annak referenciaja.

- **Metódusok**

- **pickupPump(Mechanic m):** Ha van rajta pumpa, akkor azt a paraméterül kapott játékosnak odaadja.
- **void step():** minden körben, ha a bemeneti csővén van víz, akkor a Game osztályon keresztül ad egy pontot a szerelőknek.
- **void addWater():** Víz kerül a ciszternába, pontot kapnak a szerelők.
- **void addNeighbor(Pipe p):** meghívja a szülő (**Field**) osztály **addNeighbor(Field f)** metódusát, paraméterként átadva neki p-t, majd az **input** tagváltozóját beállítja p-re.
- **void removeNeighbor(Field f):** meghívja a szülő osztály **removeNeighbor(Field f)** metódusát, paraméterként átadva neki f-et, majd az **input** tagváltozóját beállítja null-ra.

### 3.2.2 Field

- **Felelősség**

Absztrakt osztály, melynek funkciója hogy összegyűjtse azokat az objektumokat, amire a játékosok rá tudnak lépni.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **List<Player> players:** Számontartja, hogy milyen játékosok állnak rajta
- **List<Field> neighbors:** Számontartja a szomszédos Field-eket

- **Metódusok**

- **void addPlayer(Player p):** Felvesz egy új játékost a players tömbbe. Ez a játékos most már ezen a mezőn áll.
- **void removePlayer(Player p):** A paraméterül kapott játékost kiveszi a players tömbből, ezzel azt jelezve hogy a játékos lelépett erről a mezőről.
- **void accept(Player p):** Amennyiben léphet rá a paraméterül kapott játékos, lépteti őt és hozzáadja a players listához.
- **void changePumpDirection(Pipe in, Pipe out):** A mezőn nem lehet a pumpálási irányt állítani.
- **void addNeighbor(Field f):** A paraméterül kapott mezőt hozzáveszi a szomszédok listájához, tehát egy új objektummal szomszédos a jelenlegi mező
- **void removeNeighbor(Field f):** a paraméterben kapott mezőt eltávolítja a szomszédok listájából, tehát ez a szomszéddal megszünt a kapcsolata
- **void placePump(Mechanic m):** Nem tud lerakni a paraméterként kapott játékos erre a mezőre pumpát.
- **void pickupPump(Mechanic m):** Nem tud felvenni erről a mezőről a paraméterként kapott játékos pumpát.
- **void repair():** Ezt a mezőt nem lehet megjavítani.
- **void break():** Ezt a mezőt nem lehet tönkretni.

### 3.2.3 Game

- **Felelősség**

A Game osztály felelőssége a játék mindenkorai állapotának a tárolása, irányítása. Ebbe bele kell érteni a játékosok listájának jegyzését, a mezők számértartását, az éppen soron lévő játékos és a csapatok pontjainak számértartását.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **List<Field> fields:** A mezőket számértartó lista
- **List<Player> players:** A játékosokat számértartó lista
- **Player activePlayer:** A jelenleg aktív játékos referencia
- **int saboteurPoints:** A szabotör játékosok pontja
- **int mechanicPoints:** A szerelő játékosok pontja
- **int actionNumber:** Ebben a körben még hány akciót tud végrehajtani az aktuális játékos

- **Metódusok**

- **void play():** A játékot indító metódus.
- **void end():** Megállítja a játékot.
- **void endTurn():** A kör váltást megkezdő metódus. Kör váltáskor a periodikus pályaelemek lépnek egyet, az állapotos metódusok állapotot változtatnak, majd a soron következő játékos lesz aktív.

- **void breakField()**: Ezt a metódust meghívva a jelenlegi aktív játékos tönkretesz az éppen aktuális helyzetének megfelelő mezőt (ha olyan mezőn áll, amit elronthat, de ennek eldöntése a mező felelőssége).
- **void repairField()**: Ezt a metódust meghívva a jelenlegi aktív karakter megjavítja az éppen aktuális helyzetének megfelelő mezőt (ha olyan mezőn áll, amit meg lehet javítani, de ennek eldöntése a mező felelőssége).
- **void place()**: Ezt a metódust meghívva az aktív karakter lehelyezi a nála lévő pumpát (ha olyan mezőn áll, amin ez lehetséges, de ennek eldöntése a mező felelőssége).
- **void pickup()**: Ezt a metódust meghívva az aktív karakter felvesz egy pumpát (ha olyan mezőn áll, amin ez lehetséges, de ennek eldöntése a mező felelőssége).
- **void actionTaken()**: Csökkenti a körben hátralévő akciók számát.
- **void addMechanicPoints()**: Pontot ad a szerelőknek.
- **void addSaboteurPoints()**: Pontot ad a szabotöröknek.
- **void changePumpDirection(Pipe in, Pipe out)**: Meghívja az aktív játékos changePumpDirection(Pipe in, Pipe out) függvényét.
- **void movePipe(Pipe p, Field oldEnd, Field newEnd)**: Átállítja a kiválasztott cső (p) egyik végét (oldEnd) egy másik helyre (newEnd).

### 3.2.4 Mechanic

- **Felelősség**

A Player osztályhoz képest plusz felelőssége, hogy számoltartja, hogy van-e a karakternél pumpa, és ezt le tudja rakni. Emellett javítani is tud mezőket.

- **Ősosztályok**

Player

- **Interfészek**

Nincs

- **Attribútumok**

- **Pump carriedPump**: A szerelőnél lévő pumpa.

- **Metódusok**

- **void place()**: Ha van nála pumpa, megpróbálja letenni a mezőjére.
- **void pickup()**: Ha nincs nála pumpa, megpróbál felvenni egyet a mezőről, amin éppen áll.
- **void doPickup(Pump p)**: A paraméterként kapott pumpát felveszi, és csökkenti a körben hátralévő akciók számát 1-el.
- **void doPlace()**: Ha eddig volt pumpája, most már nem lesz, lerakja a jelenlegi mezőjére, úgy, hogy amin áll, azt 2 csővel és közöttük 1 pumpával helyettesíti és csökkenti a körben hátralévő akciók számát 1-el.
- **void repairField()**: Megpróbálja megjavítani azt a mezőt, amin éppen áll.

### 3.2.5 Periodic

- **Felelősség**

Lehetővé teszi hogy a hálózat objektumai sorban meghívodjanak egy-egy körben. Ezalatt, minden egyik típusától függően definiál egy viselkedést. Ezen viselkedések összessége alkotja a hálózat folyamát.

- **Metódusok**

- **void step():** minden objektum ami implementálja, ebbe írja le a sajátos viselkedését.

### 3.2.6 Pipe

- **Felelősség**

Felelőssége, hogy szamontartsa, van-e benne viz. Azt is szamontartja, hogy el van-e törve és ha igen és víz van benne, a kör végén ad pontot a szabotőröknek.

- **Ősosztályok**

Field

- **Interfészek**

Periodic, Stateful

- **Attribútumok**

- **bool oldWaterState:** azt jegyzi le hogy ha van benne víz a legutolsó kör kezdete óta.
- **bool newWaterState:** minden körben az updateState() hívásra ennek a változónak az értékét átmásolja az oldWaterState változóba.
- **bool isBroken:** True, ha el van törve a cső, false, ha nincs.

- **Metódusok**

- **void step():** A cső nem folyatja a vizet.
- **void accept(Player p):** Ha nem áll rajta más játékos és a paraméterben kapott játékos egy mellette lévő mezőn áll, akkor önmagára lépteti őt és hozzáadja a players listához.
- **void changeState():** Ha különbözik a két állapot (newWaterState) és (oldWaterState), akkor az oldWaterState megkapja a newWaterState értékét. Ezután, ha az oldWaterState igaz és a cső lyukas, a szabotőrök kapnak pontot.
- **void repair():** Az isBroken értéke hamis lesz és amennyiben igaz volt, csökkenti a hátralévő akciók számát 1-el.
- **void break():** Az isBroken értéke igaz lesz és amennyiben hamis volt, csökkenti a hátralévő akciók számát 1-el.
- **void placePump(Mechanic m):** A paraméterül kapott szerelő játékoson meghívja a doPlace() metódust (lerakatja vele a pumpát).
- **void flowWater(Cistern c):** A paraméterül kapott ciszternán meghívja az addWater() függvényt, ha az oldWaterState igaz ("folyatja bele a vizet").
- **void giveWater(Pump p):** Ad a paraméterül kapott pumpa tartályába vizet, ha oldWaterState igaz és eltünteti onnan a vizet, ha az oldWaterState hamis.

### 3.2.7 Player

- **Felelősség**

Absztrakt osztály, mely definiálja a Saboteur és a Mechanic közös viselkedését: mozgás a hálózaton és annak nyilvántartása, hogy milyen mezőn áll.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

- **Field activeField:** Azt jelzi hogy melyik mezőn áll abban a pillanatban.

- **Metódusok**

- **void moveToField(Field f):** A paraméterként kapott mezőre megpróbál átlépni.
- **void breakField():** Nem tud mezőt eltörni.
- **void repairField():** Nem tud mezőt megjavítani.
- **void pickup():** Nem tud felvenni a mezőről dolgokat.
- **void place():** Nem tud lerakni a mezőre dolgokat.
- **void swapField(Field f):** A paraméterként kapott mezőre átlép. Kiveszi magát (removePlayer()) a jelenlegi mezőjének listájából majd csökkenti a körben hátralévő akciók számát 1-el.
- **void changePumpDirection(Pipe in, Pipe out):** Megpróbálja átállítani a jelenlegi aktív mezőjén a bemeneti és kimeneti csöveket (ha olyan mezőn áll, amin ez lehetséges, de ennek eldöntése a mező felelőssége).

### 3.2.8 Pump

- **Felelősség**

A jatekszabalyok szerinti pumpa funkcioit valositja meg. minden kör végén először ha a tartályában van víz, a kimenetén is lesz, ha nincs akkor a kimenetén sem lesz. Ezután ha a bemenetén van víz, a tartályába beszívja azt, ha nem volt, akkor a tartálya üres lesz. Lehetővé teszi, hogy átállítsák a játékosok és minden kör végén van egy kis esély arra, hogy elromlik.

- **Ősosztályok**

Field

- **Interfészek**

Periodic

- **Attribútumok**

- **int hasStoredWater:** számontartja, hogy van-e a víztartályban víz.
- **bool isBroken:** számontartja, hogy a pumpa elromlott-e.
- **Pipe input:** A bemeneti csöve
- **Pipe output:** A kimeneti csöve

- **Metódusok**

- **void step():** Kis eséllyel elromlik (isBroken true lesz). Ha nem romlott el, a következő viselkedés definiált: Ha a tartályában van víz, a kimeneti csőre tesz vizet (ilyenkor nem tűnik el a tartályból a víz). Ha a tartályában nincs víz, a kimeneti csövéről eltünteti a vizet. Majd a bementi csövéről megpróbál beszívni vizet a tartályába, ha a tartálya üres. Amennyiben a bemeneti csőben nincs víz, a tartály üres lesz, ha van benne víz, a tartálya tele lesz.
- **void changePumpDirection(Pipe in, Pipe out):** Beállítja a bemeneti csövet az in paraméter által kapott csőre, a kimeneti csövet az out paraméter által kapott csőre,

amennyiben ez a két paraméter nem egyezik meg (a bemeneti és kimeneti csövek az **input** és **output** tagváltozók, melyeket a **Pump** a **Field** osztályból örököl).

- **void repair():** Ha eddig el volt romolva, akkor megjavul és csökkenti a körben hátralévő akciók számát 1-el.
- **void setWater(boolean b):** A víztartály töltöttsgégét (hasStoredWater) a paraméterként kapott értékre beállítja.
- **removeNeighbor(Field f):** meghívja a szülő (**Field**) osztály **removeNeighbor(Field f)** metódusát, paraméterként átadva neki f-et, majd ha f megegyezik az **output** tagváltozójával, az **output** értékét **null**-ra állítja. Ha pedig f megegyezik az **input** tagváltozójával, az **input** értékét állítja **null**-ra.

### 3.2.9 Saboteur

- **Felelősség**

A Player osztályhoz képest plusz felelőssége, hogy ki tud lyukasztani csöveget.

- **Ősosztályok**

Player

- **Interfészek**

Nincs

- **Metódusok**

- **void breakField():** Megpróbálja eltörni azt a Field-et, amin áll (activeField).

### 3.2.10 Source

- **Felelősség**

Végtelen vízforrásnak számít. A kimenetén mindenkoran folyik a víz.

- **Ősosztályok**

Field

- **Interfészek**

Periodic

- **Attribútumok**

- **Pipe output:** A kimeneti csöve.

- **Metódusok**

- **void step():** A kimeneti csövére rak vizet.

- **void addNeighbor(Pipe p):** meghívja a szülő (**Field**) osztály **addNeighbor(Field f)** metódusát, paraméterként átadva neki p-t, majd az **output** tagváltozóját beállítja p-re.
- **void removeNeighbor(Field f):** meghívja a szülő osztály **removeNeighbor(Field f)** metódusát, paraméterként átadva neki f-et, majd az **output** tagváltozóját beállítja **null**-ra.

### 3.3.11 Stateful

- **Felelősség**

Az állapottal rendelkező objektumok állapotváltásáért felel.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

Nincs

- **Metódusok**

- **void changeState()**: Megváltoztatja az objektum állapotát.

### 3.3.12 Timer

- **Felelősség**

A játék köreinek léptetéséért felelős. Ismerve az összes olyan objektumot amit léptetni vagy állapotát változtatni kell, sorban az előbbieken meghívja a step() metódust. Ezután meghívja a changeState() metódust az utóbbiakon.

- **Ősosztályok**

Nincs

- **Interfészek**

Nincs

- **Attribútumok**

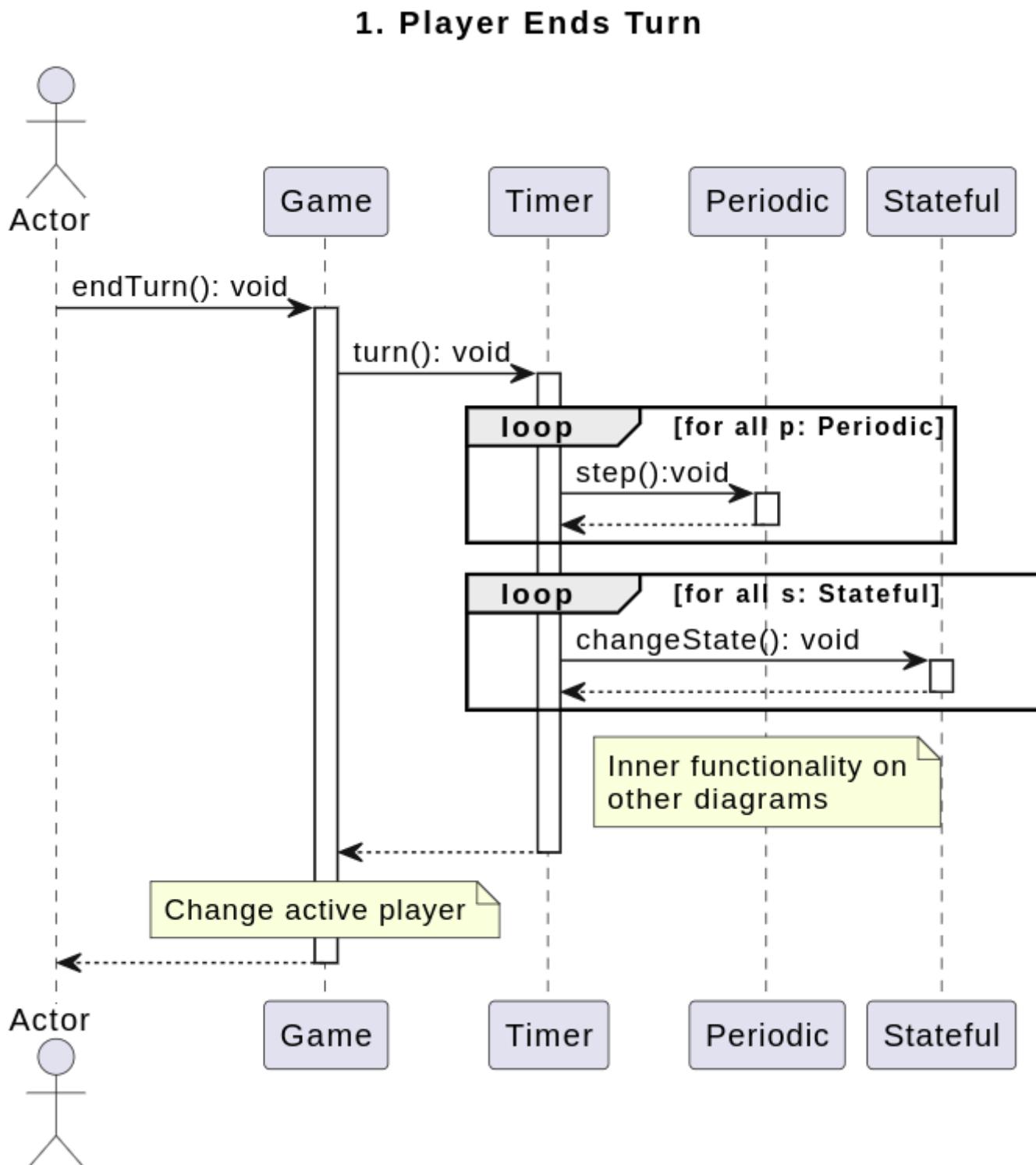
- **List<Periodic> periodics**: Számontartja az összes olyan objektumot ami megvalósítja a Periodic interface-t.
- **List<Stateful> statefuls**: Számontartja az összes olyan objektumot ami megvalósítja a Stateful interface-t.

- **Metódusok**

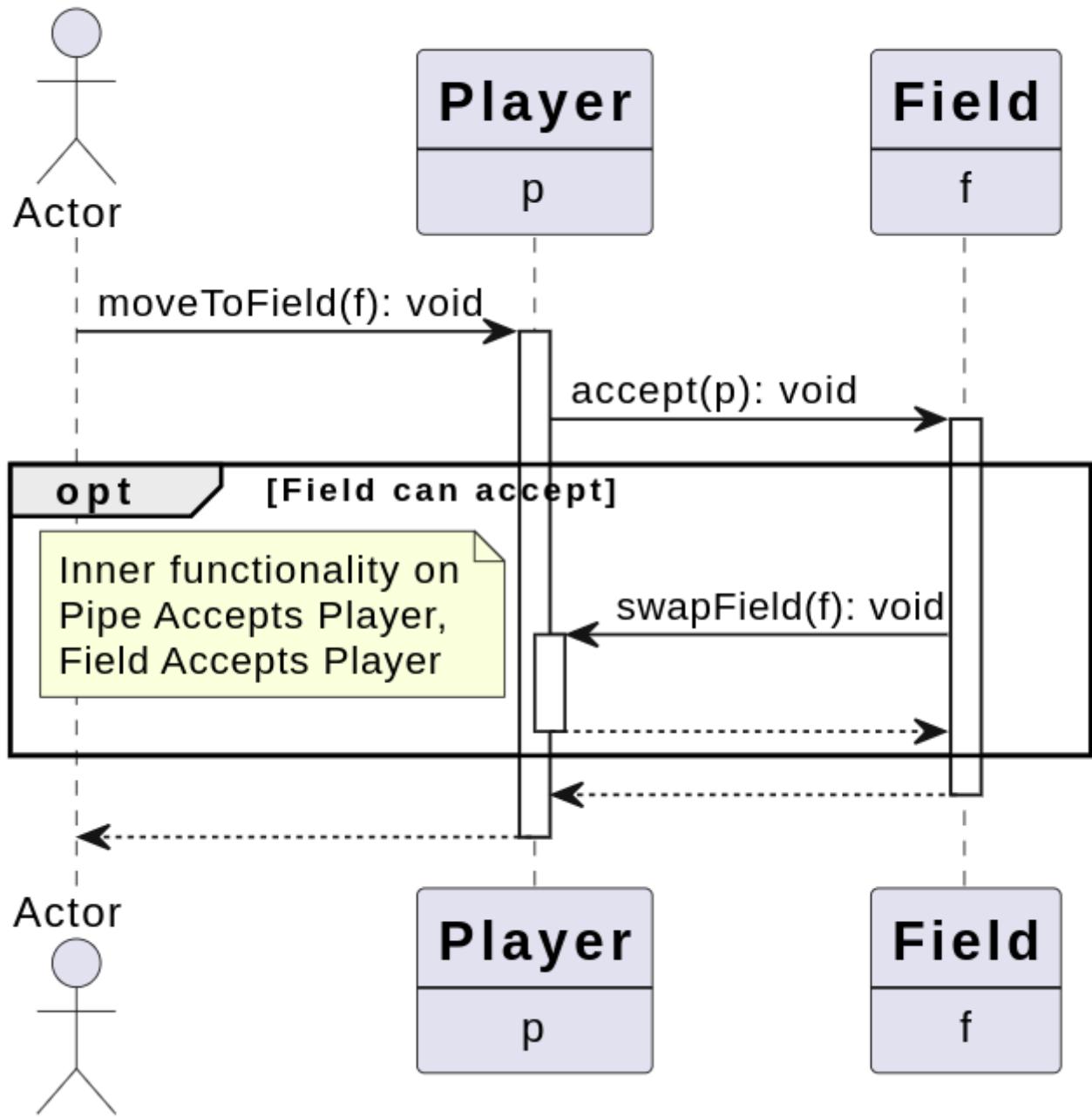
- **void turn()**: Egy ciklussal végigmegy a periodics listában lévő objektumokon és minden egyiken meghívja először a step() függvényt. Ezután megeszi ugyanezt a statefuls lista minden elemén: meghívja a changeState() metódust.

## 3.3 Szekvencia diagramok

### 1. Player Ends Turn:

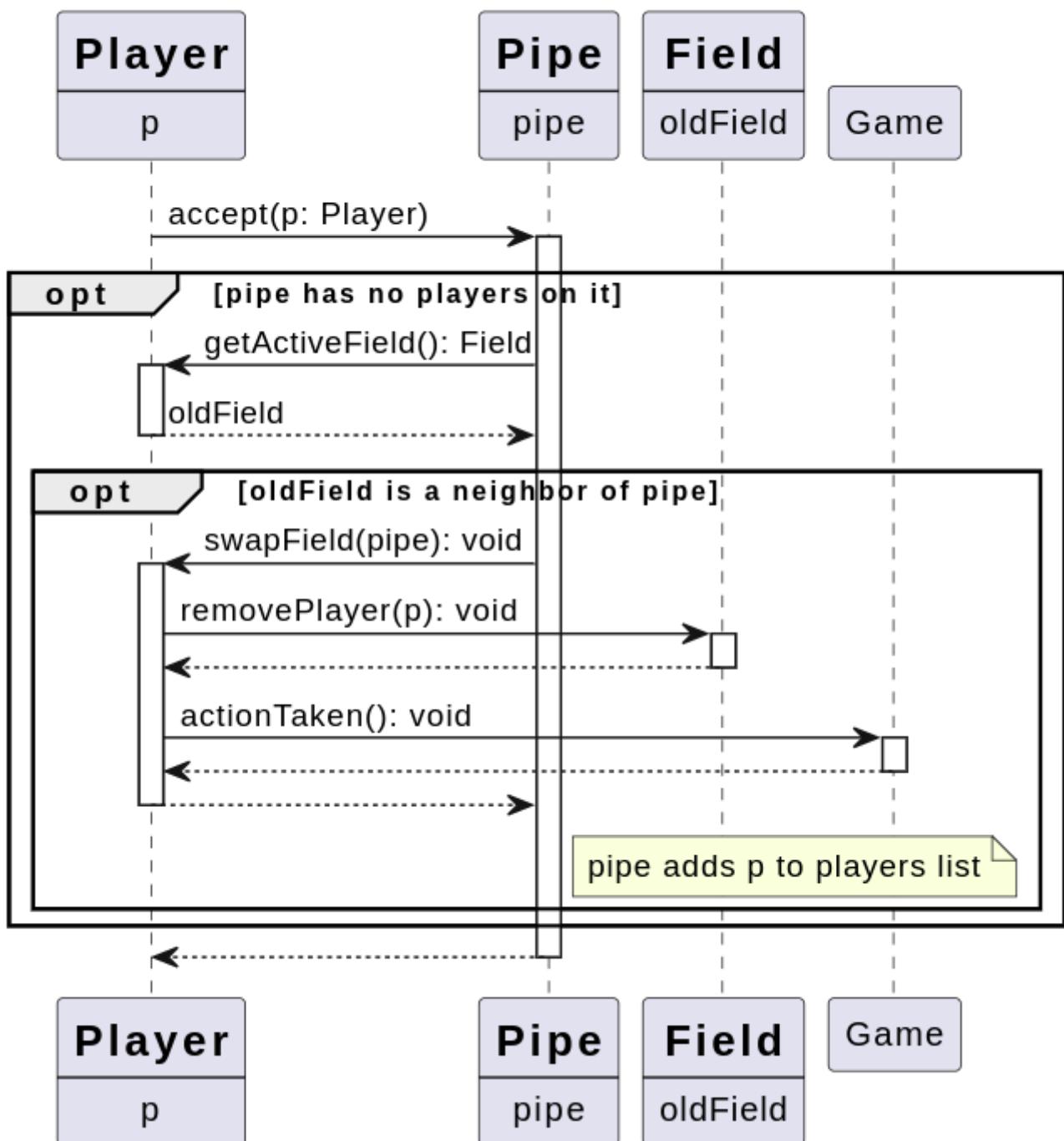
**2. Player Moves:**

## 2. Player Moves



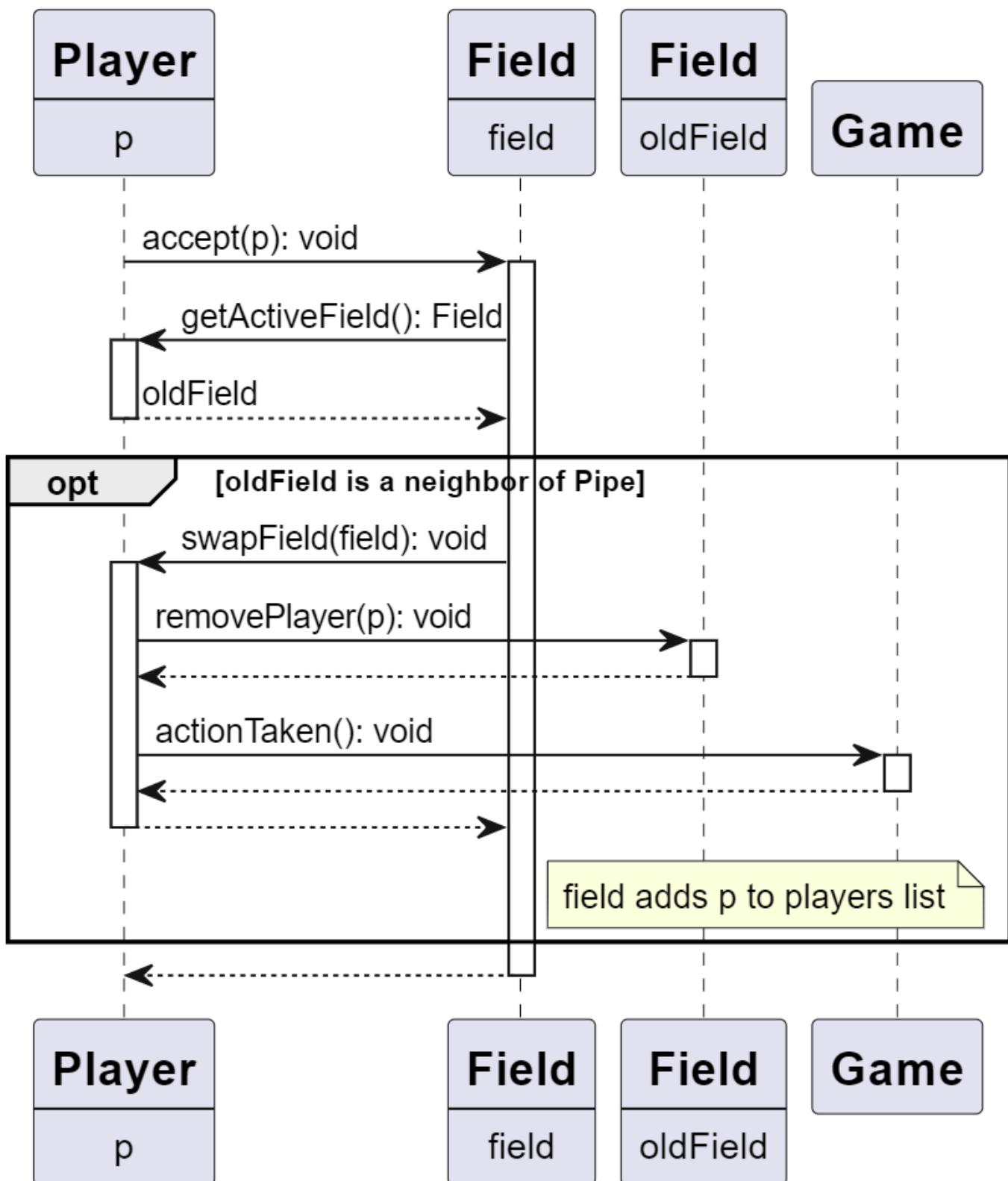
### 3. Pipe Accepts Player:

### 3. Pipe Accepts Player



### 4. Field Accepts Player:

## 4. Field Accepts Player

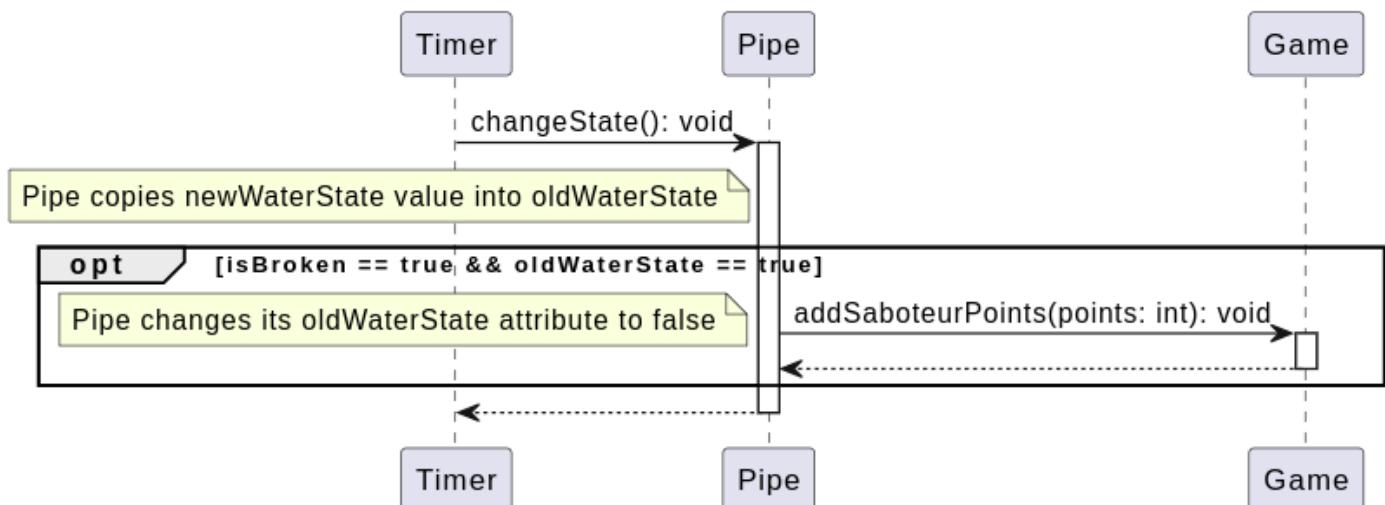


Behaviour defined in Field class. Source, Cistern, Pump inherit this.

Behaviour of Pipe class in Pipe Accepts Player

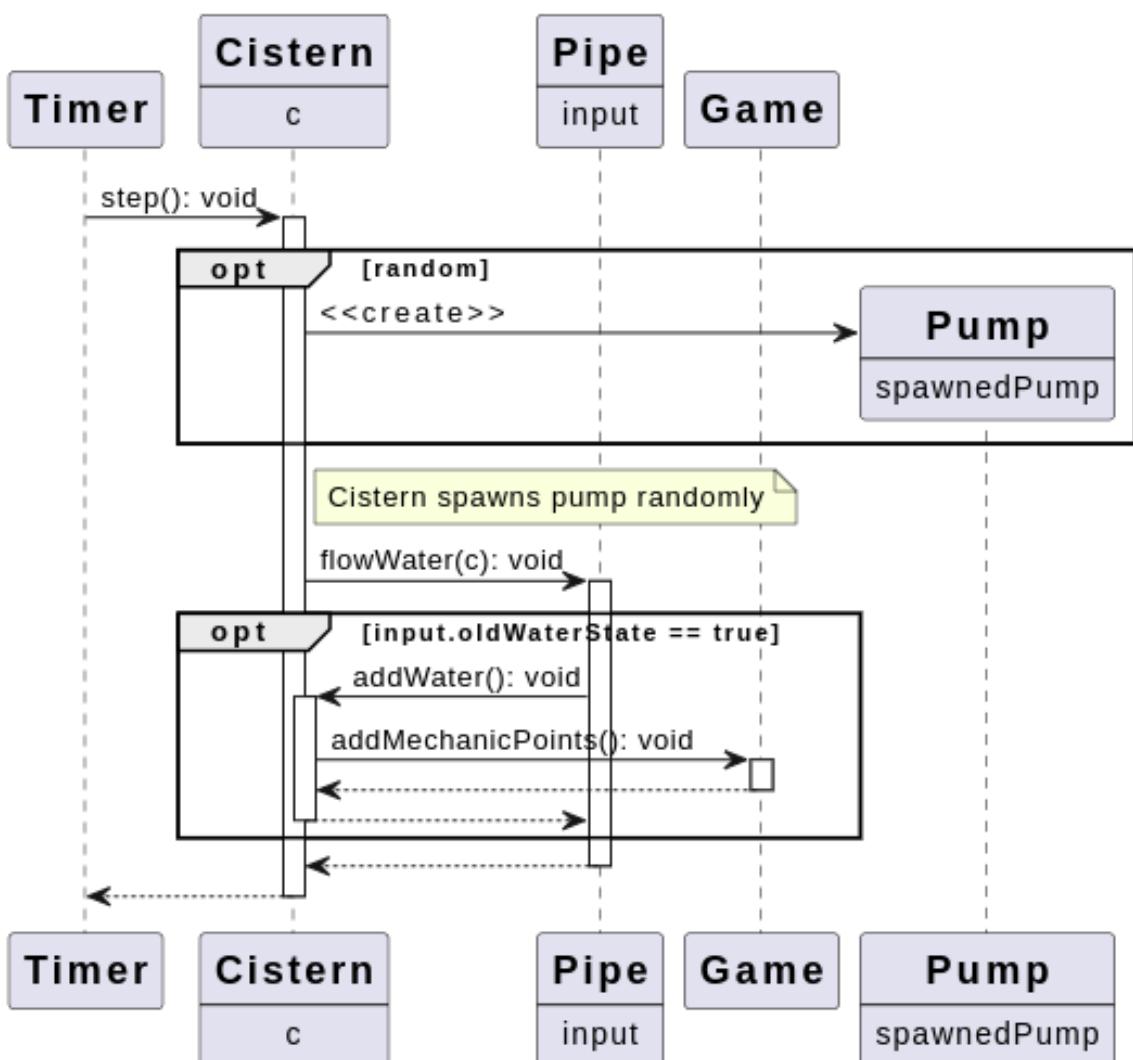
### 5. Pipe Changes State:

#### 5. Pipe Changes State

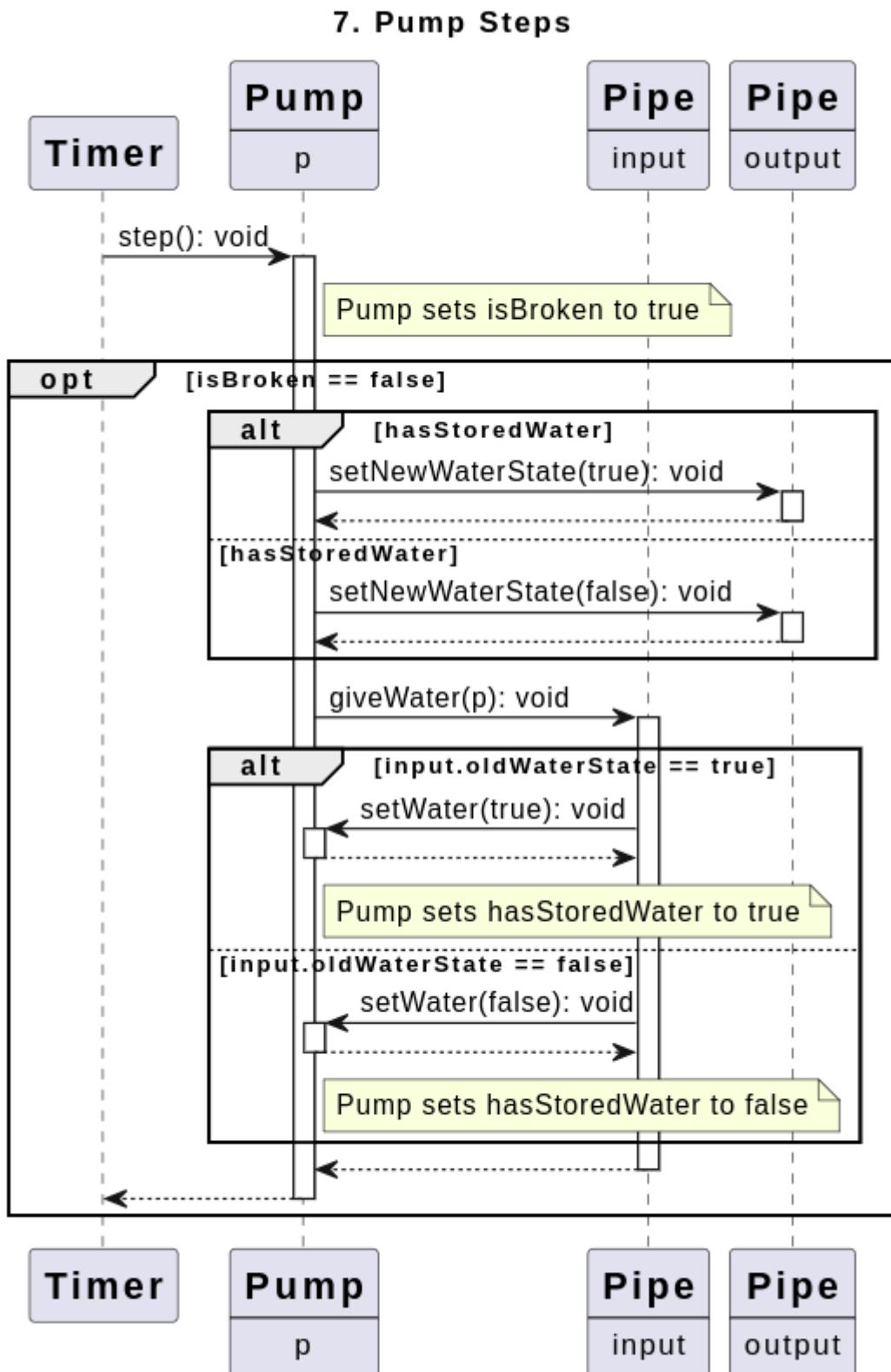


### 6. Cistern Steps:

#### 6. Cistern Steps

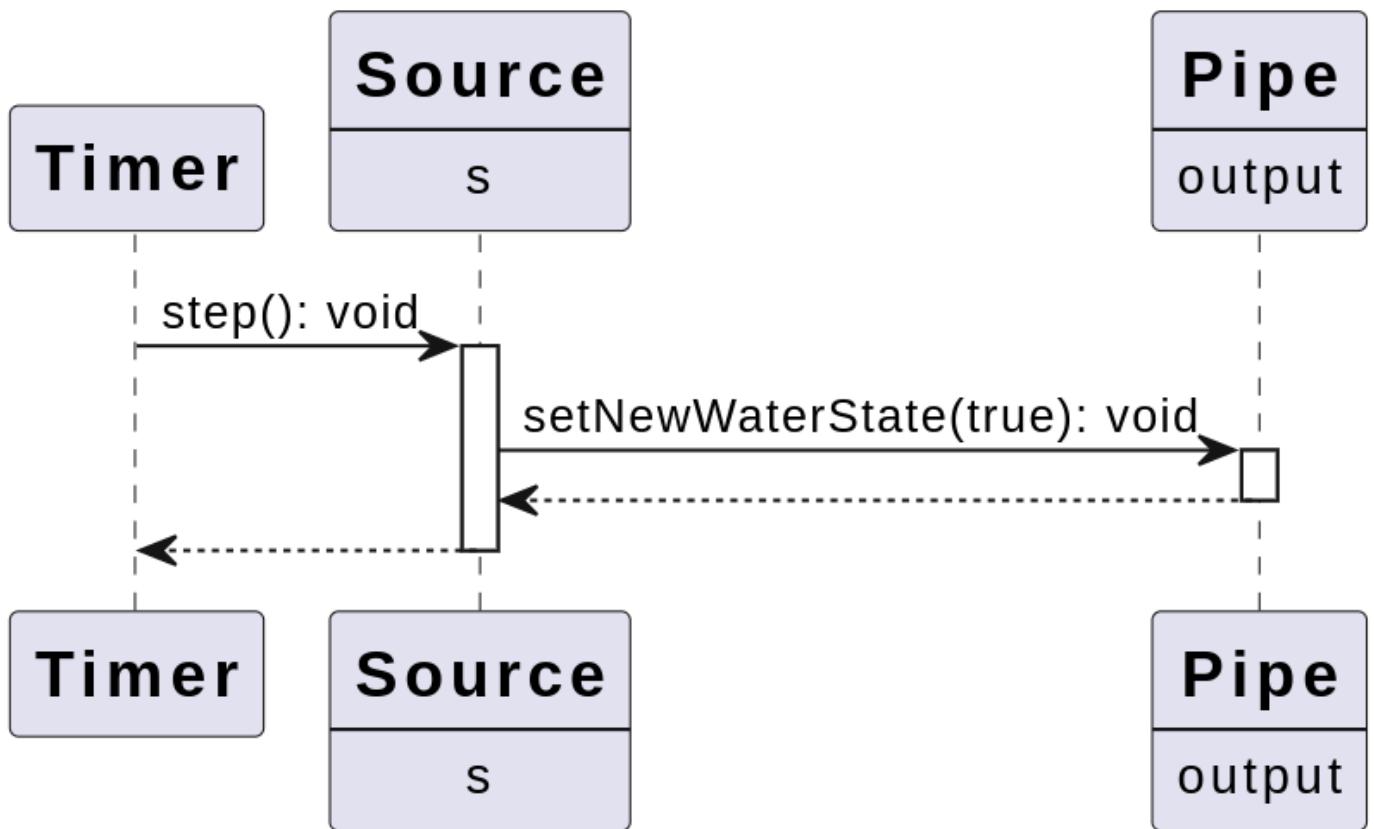


### 7. Pump Steps:



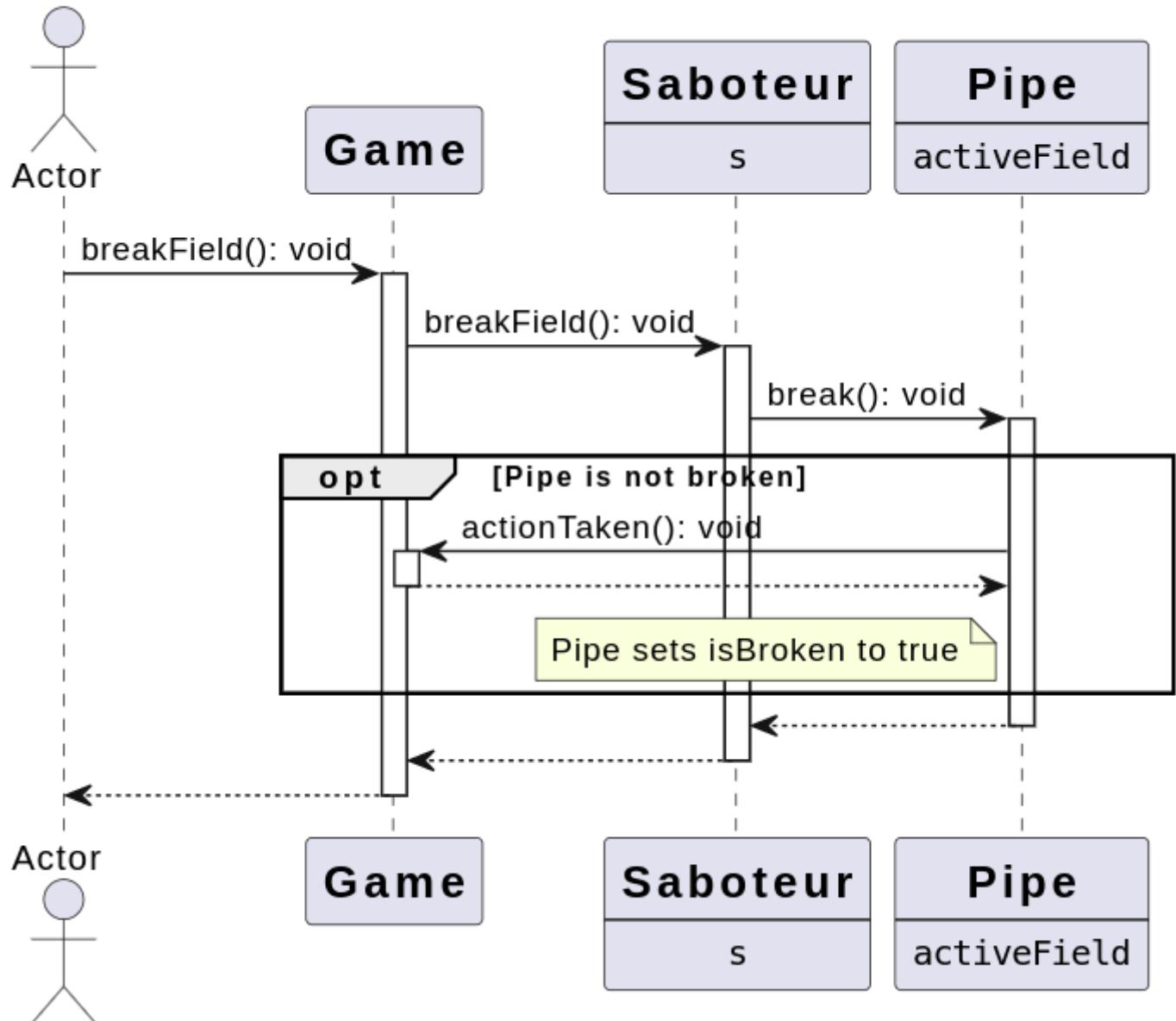
### 8. Source Steps:

## 8. Source Steps



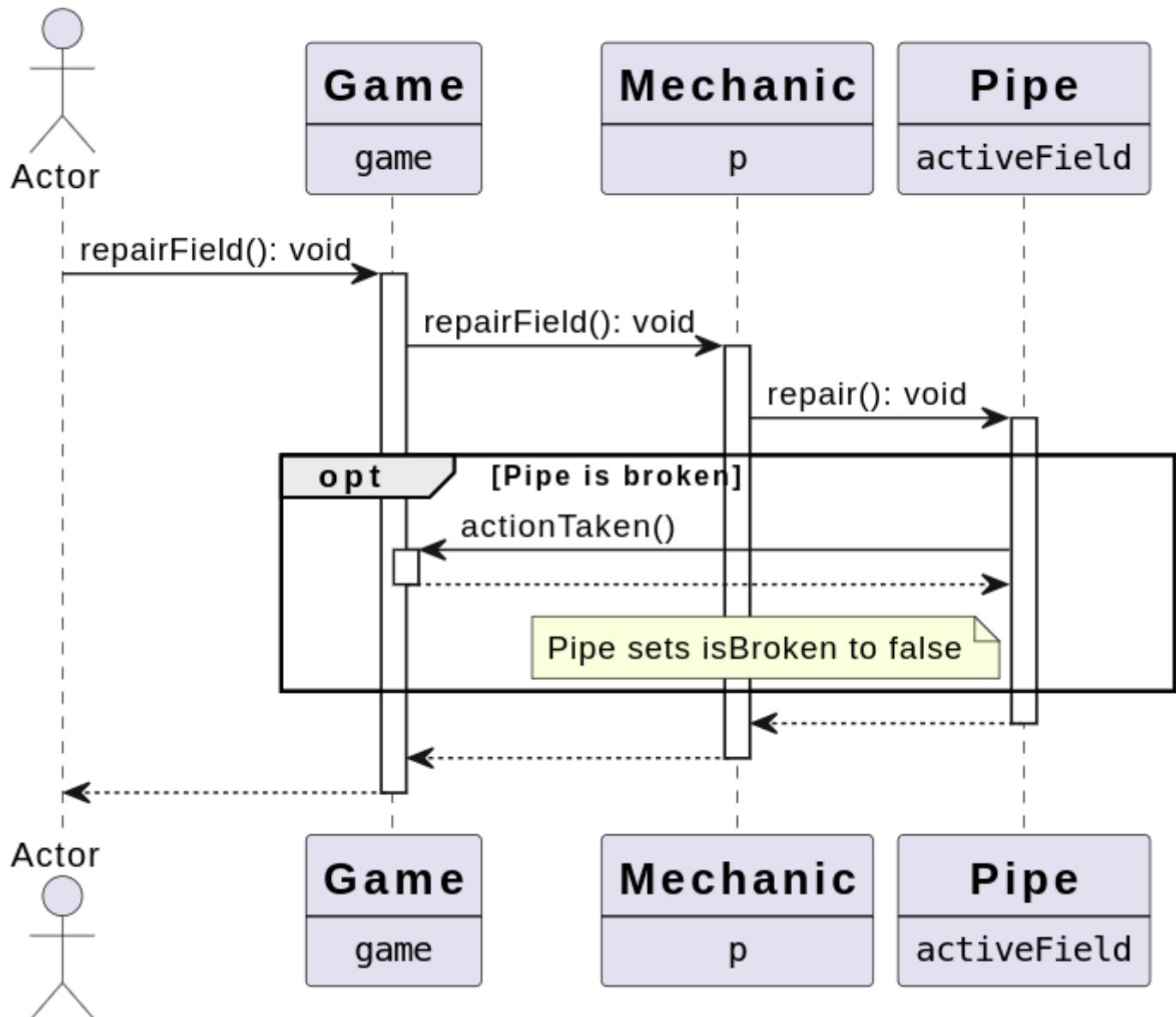
## 9. Saboteur Breaks Pipe:

## 9. Saboteur Breaks Pipe



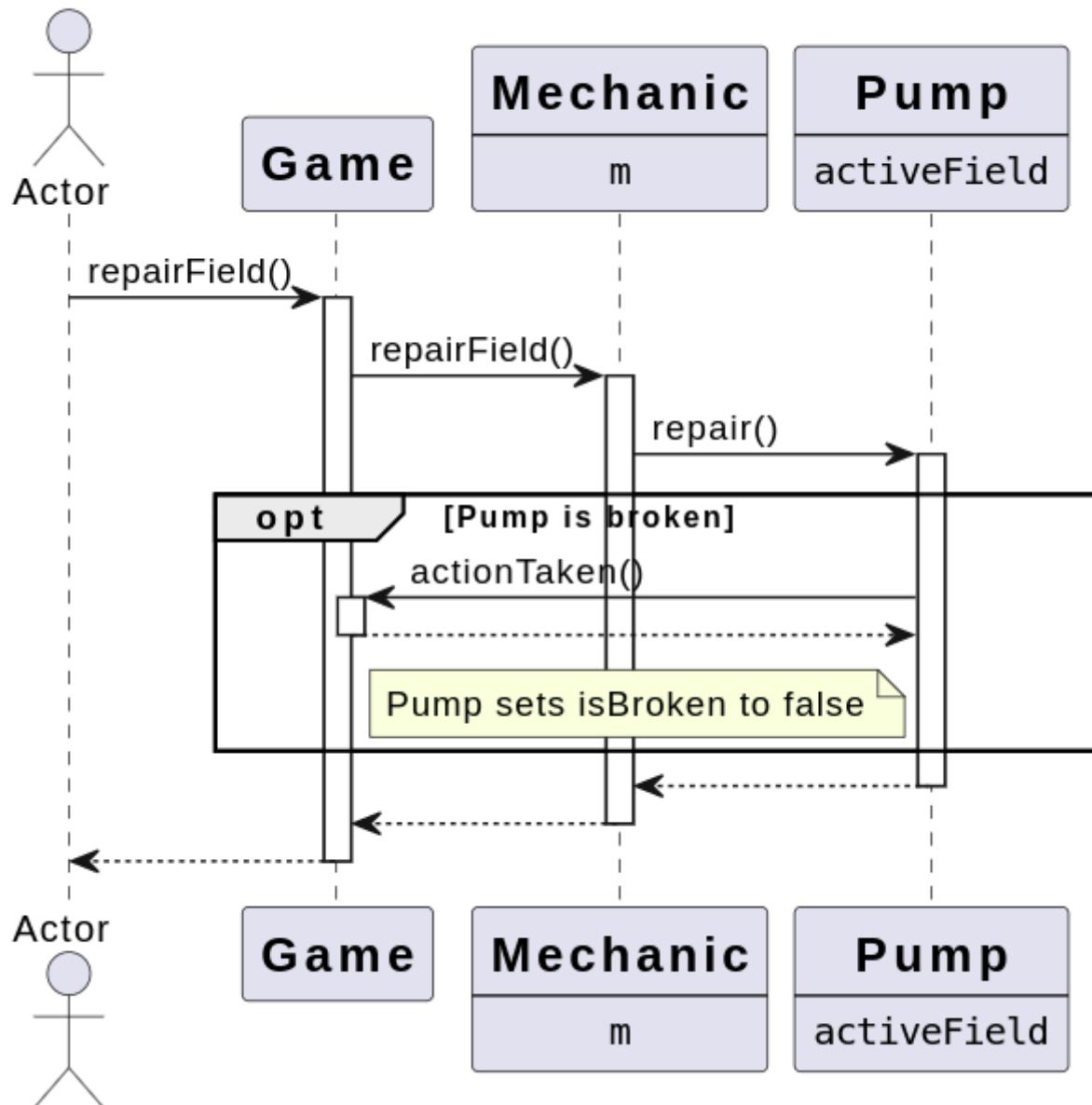
## 10. Mechanic Repairs Pipe:

## 10. Mechanic Repairs Pipe



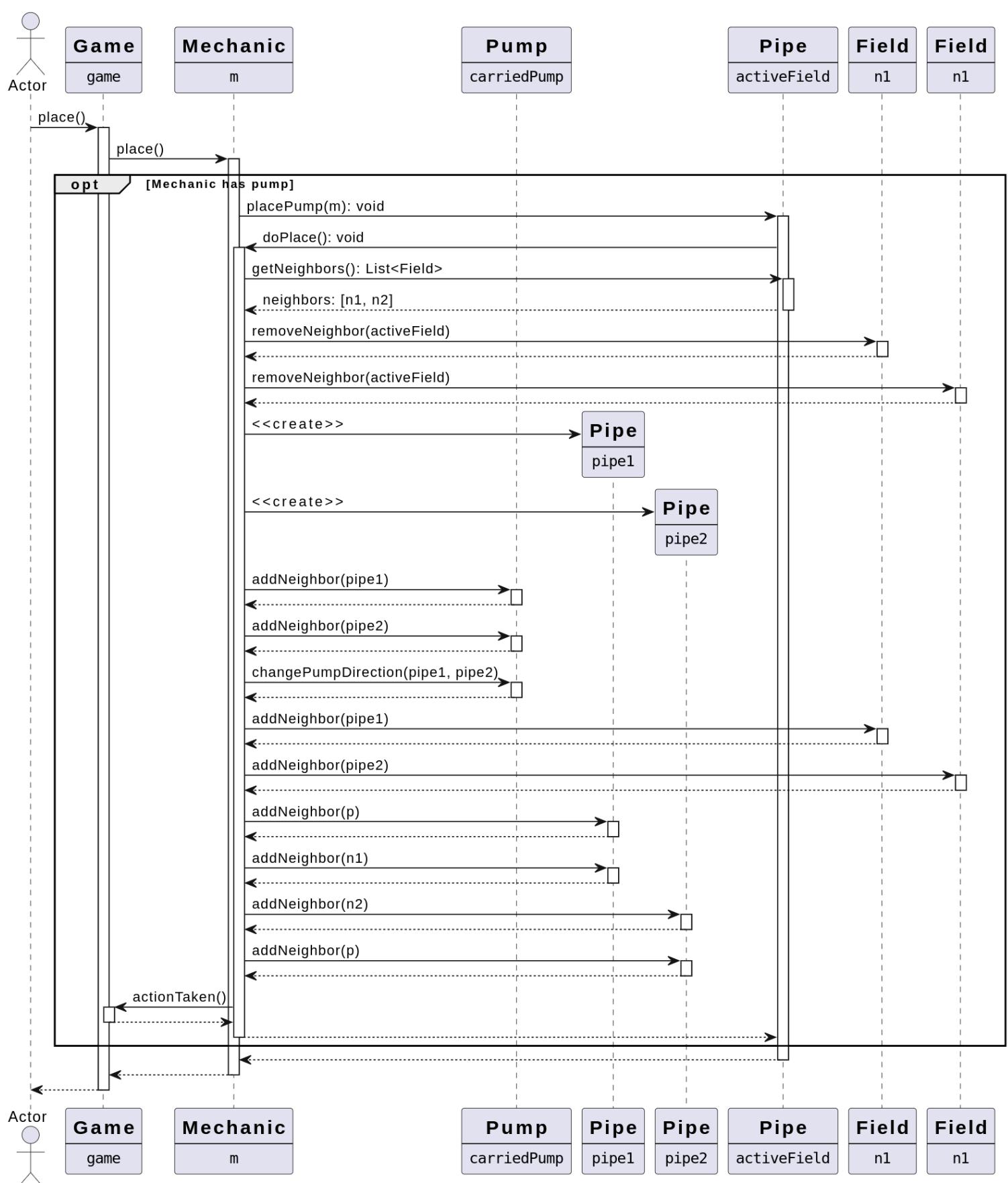
## 11. Mechanic Repairs Pump:

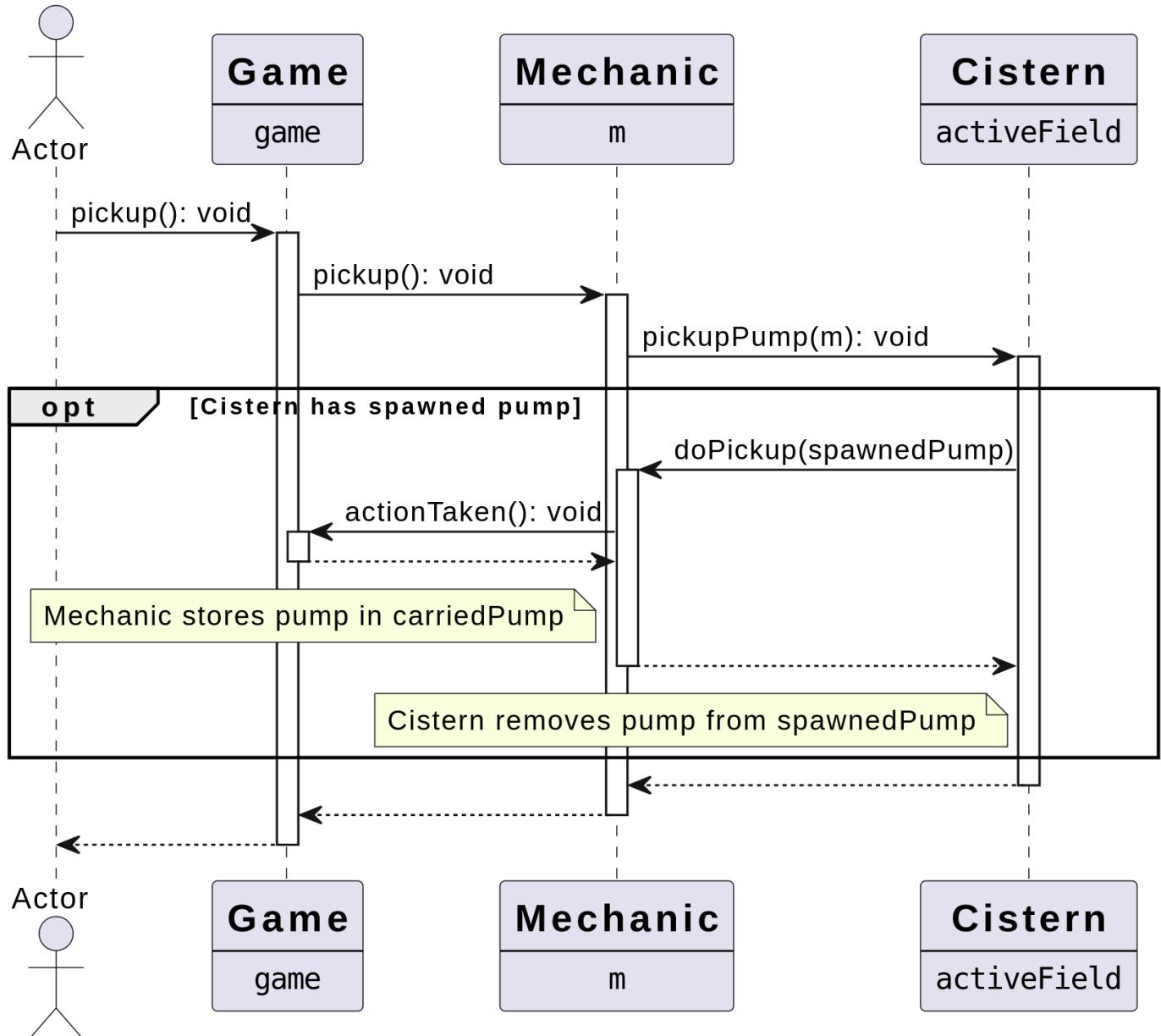
## 11. Mechanic Repairs Pump

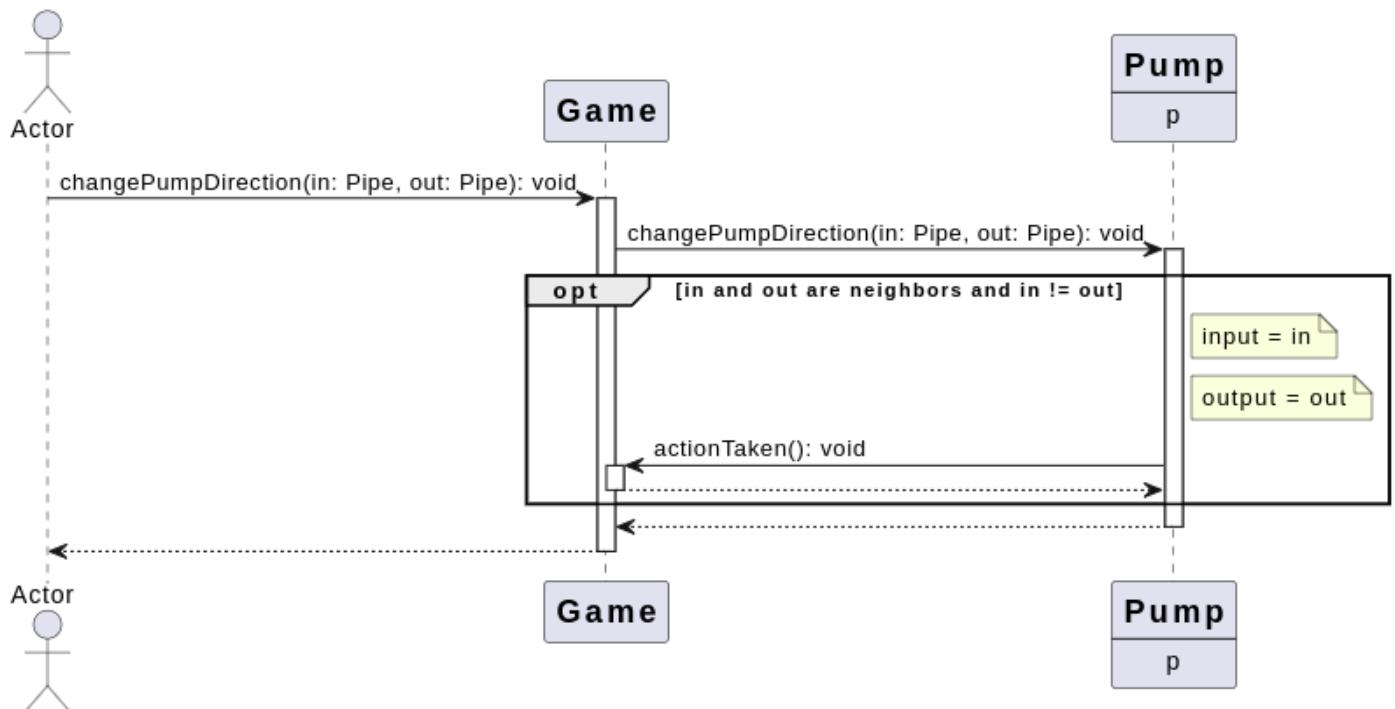
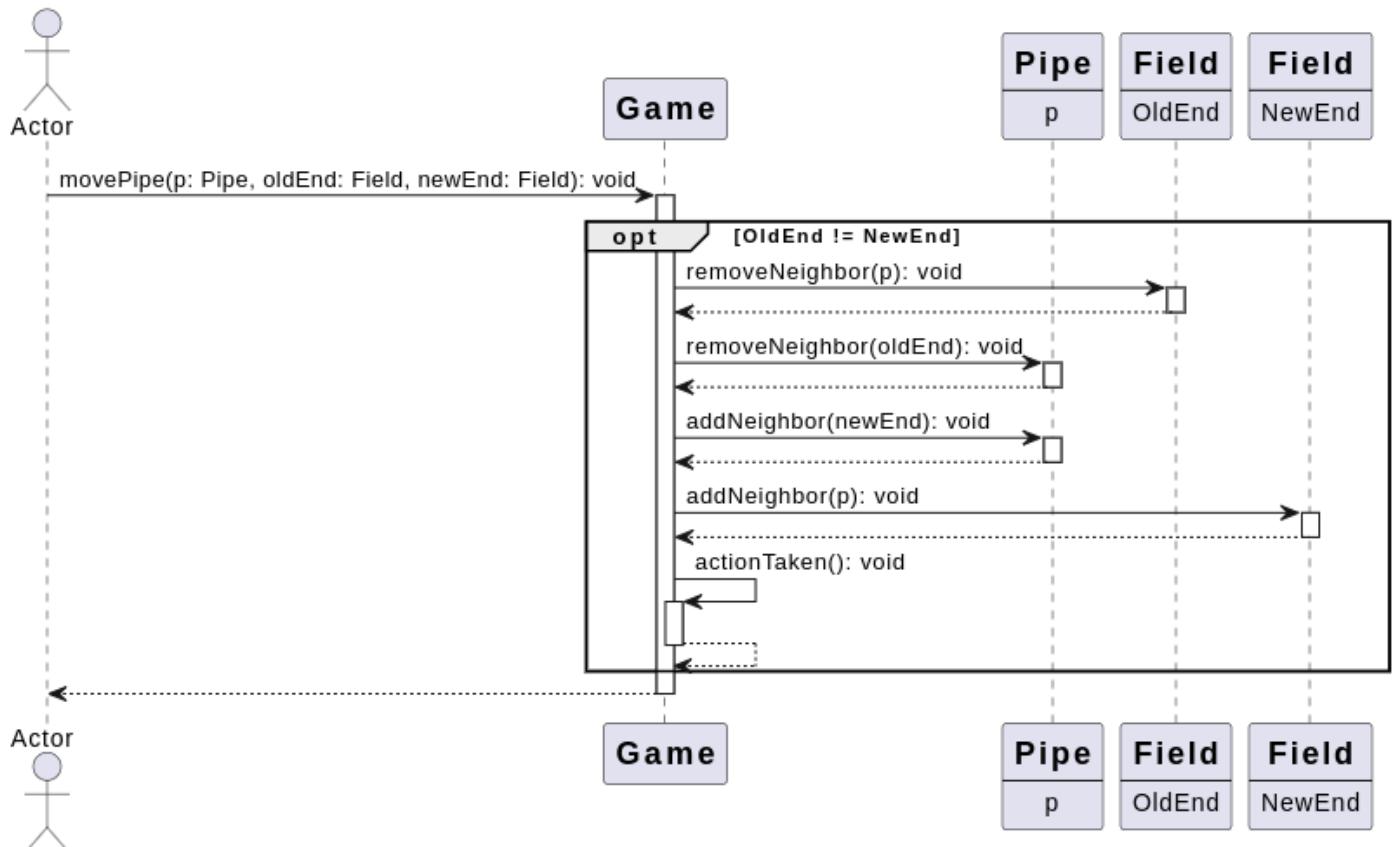


## 12. Mechanic Places Pump:

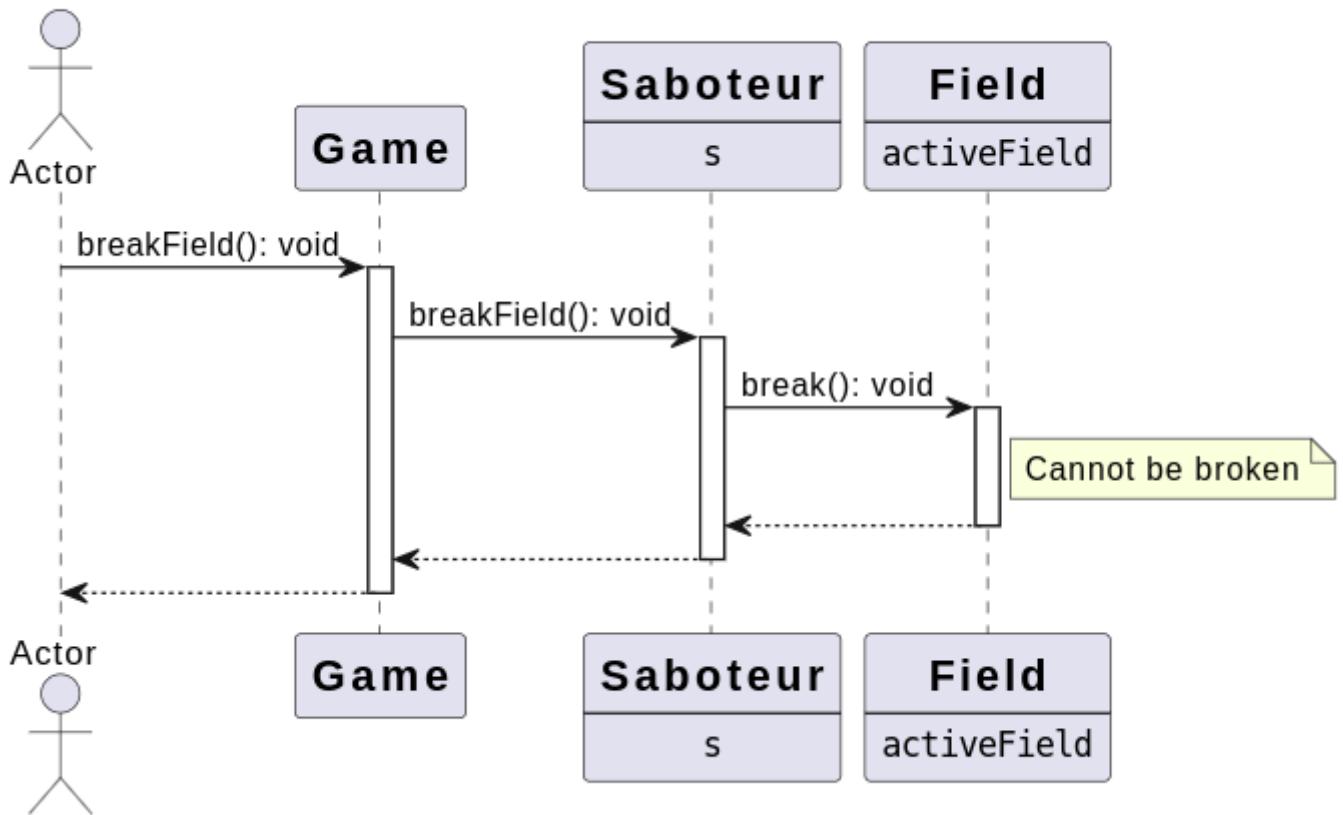
## 12. Mechanic Places Pump



**13. Mechanic Picks Up Pump:****13. Mechanic Picks Up Pump****14. Player Changes Pump Direction:**

**14. Player Changes Pump Direction****15. Transfer Pipe Endings:****15. Transfer Pipe Endings****16. Saboteur Attempts To Break (Non-Pipe) Field:**

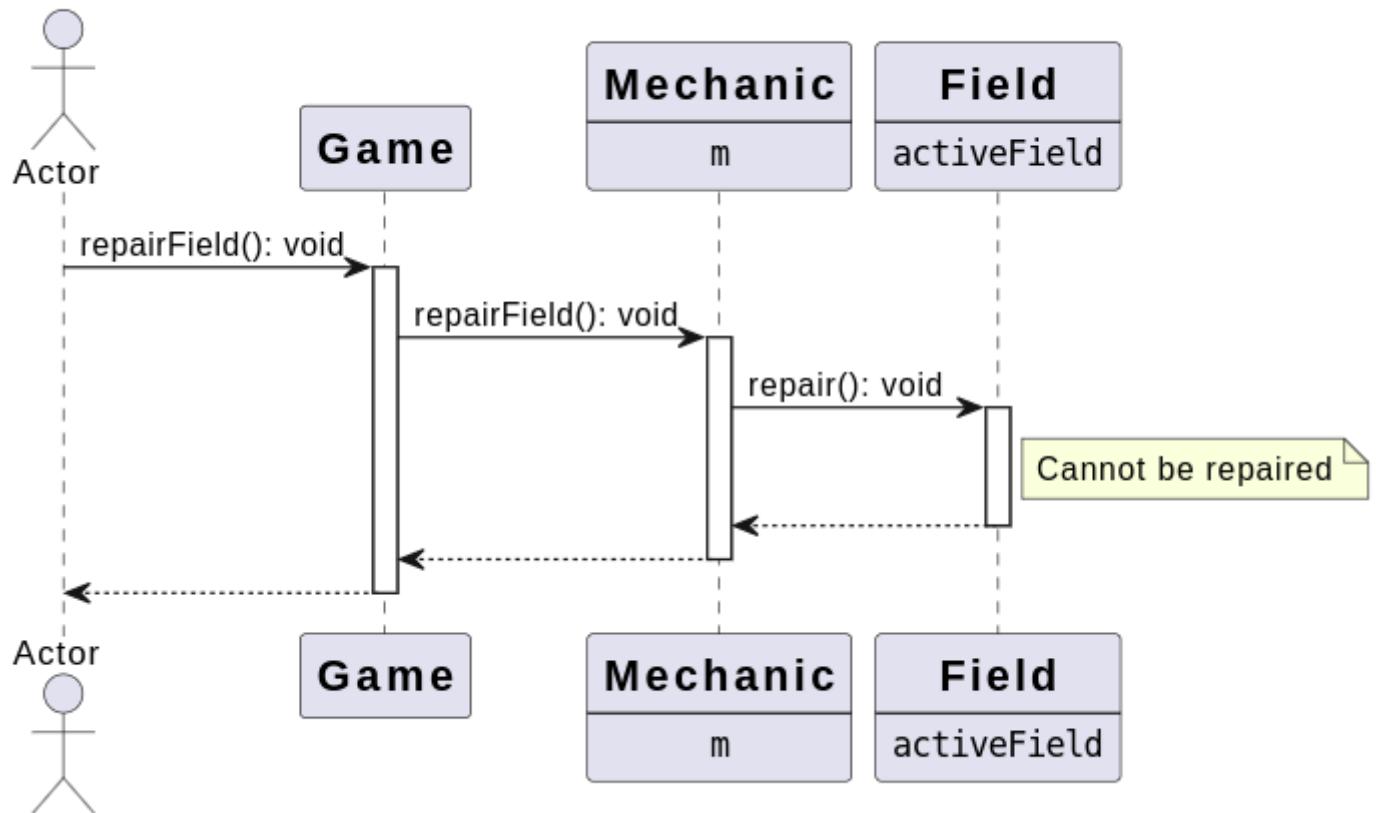
### 16. Saboteur Attempts To Break (Non-Pipe) Field



Note: Class Pipe overrides this behaviour. (See Saboteur breaks Pipe)

### 17. Mechanic Attempts To Repair (Non-Pipe/Pump) Field:

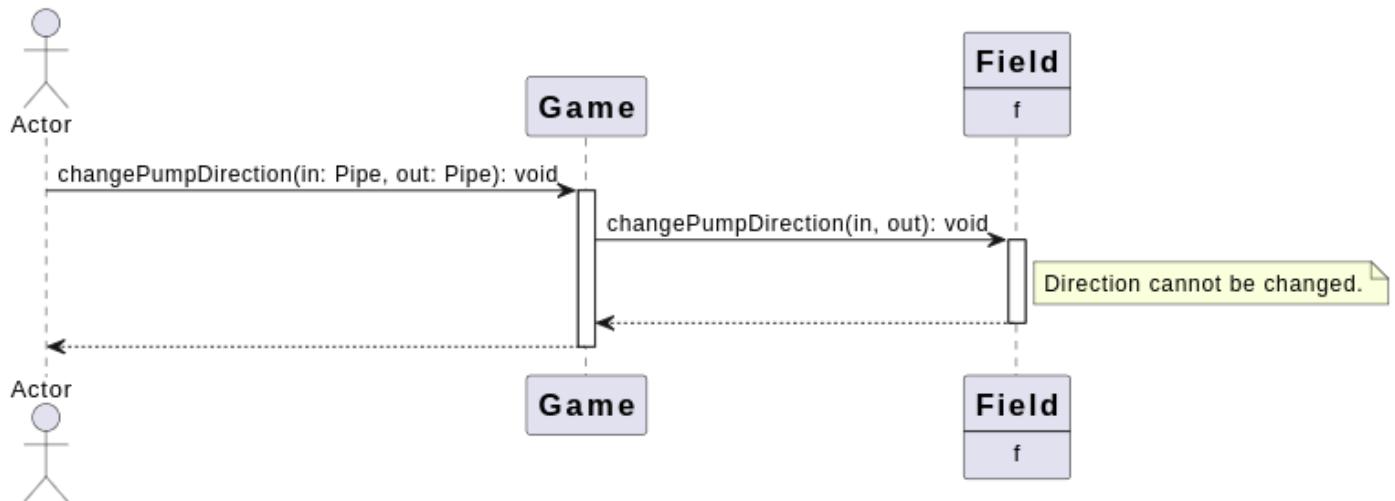
### 17. Mechanic Attempts To Repair (Non-Pipe/Pump) Field



Note: Class Pipe and Pump override this behaviour. (See Mechanic repairs Pump/Pipe)

### 18. Player Attempts To Change Pump Direction On (Non-Pump) Field:

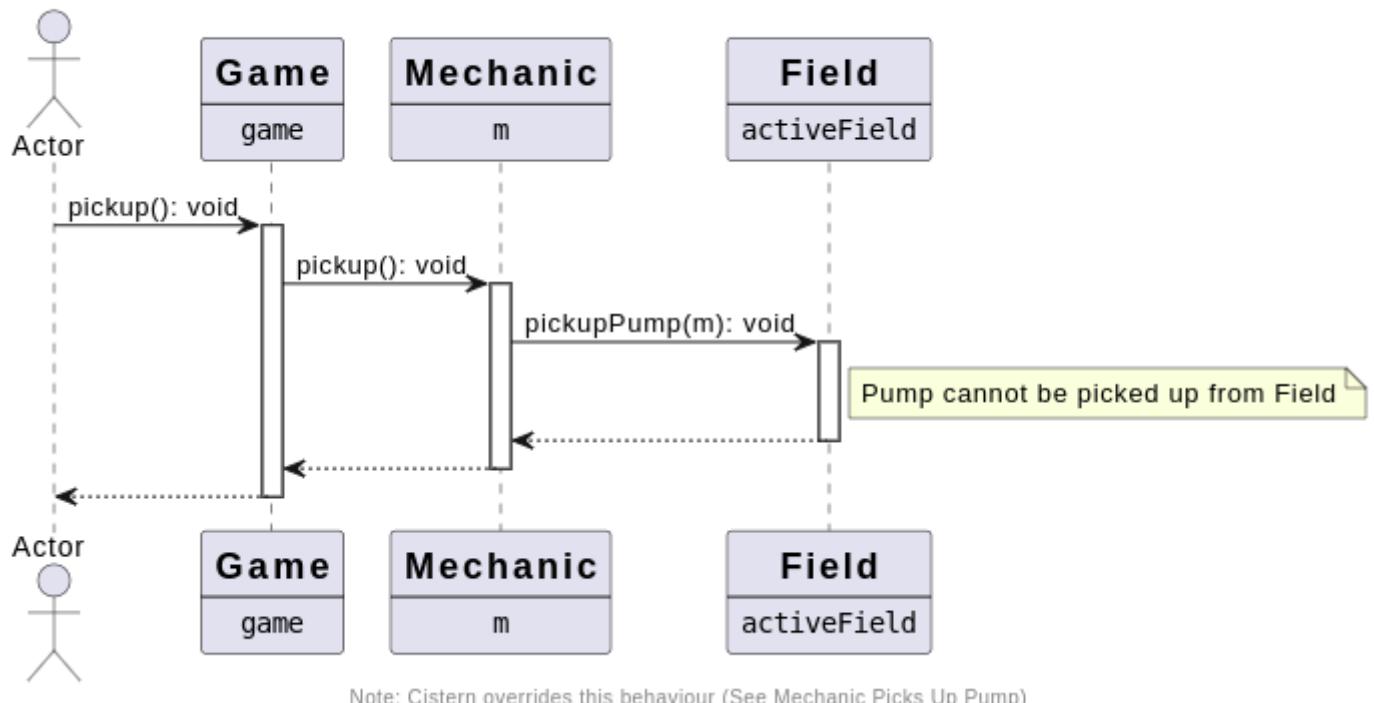
#### 18. Player Attempts To Change Pump Direction On (Non-Pump) Field



Note: Class Pump overrides this behaviour. (See Player Changes Pump Direction)

### 19. Mechanic Attempts To Pick Up Pump From (Non-Cistern) Field:

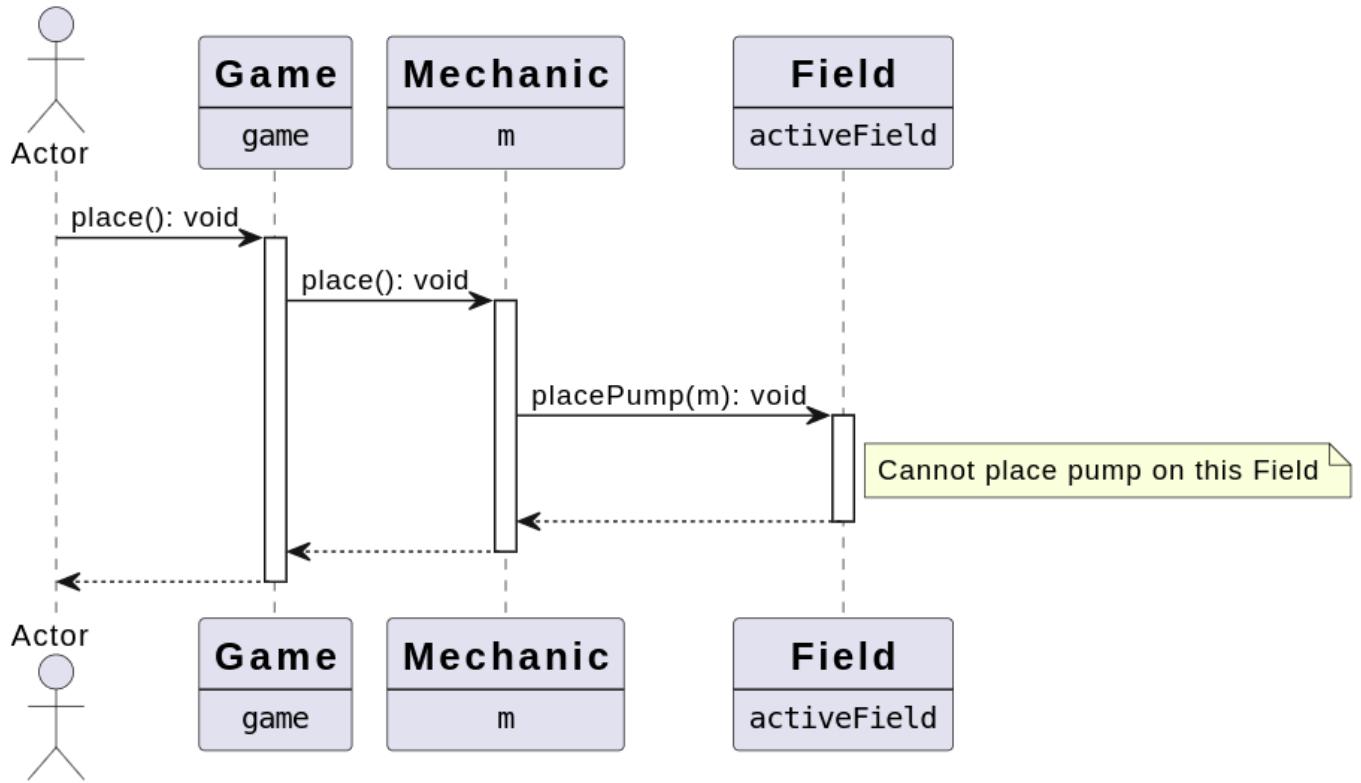
### 19. Mechanic Attempts To Pick Up Pump From (Non-Cistern) Field



Note: Cistern overrides this behaviour (See Mechanic Picks Up Pump)

### 20. Mechanic Attempts To Place Pump On (Non-Pipe) Field

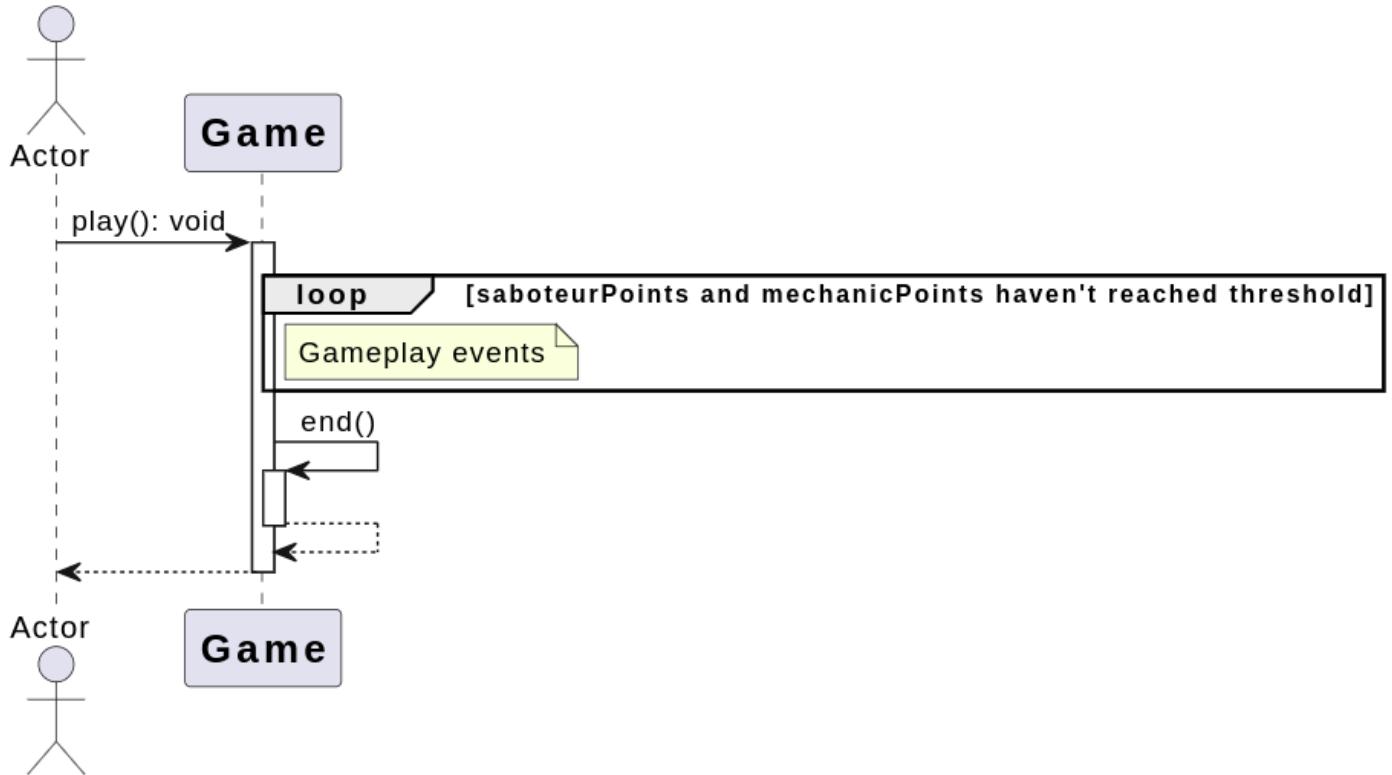
#### 20. Mechanic Attempts To Place Pump On (Non-Pipe) Field



Note: Pipe overrides this behaviour. (See Mechanic Places Pump)

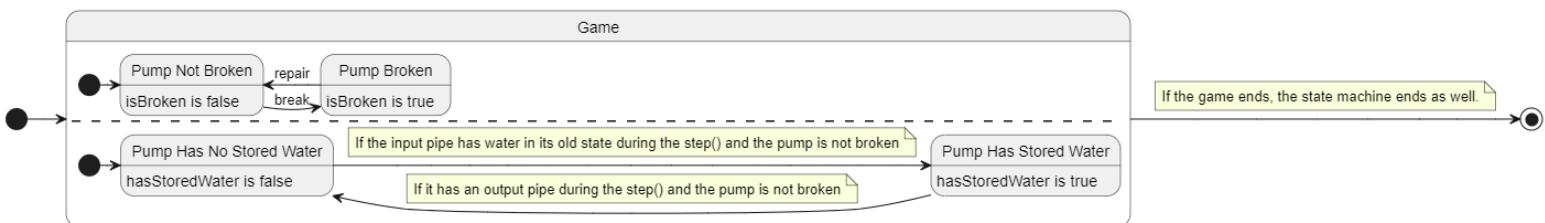
### 21. Game Starts / Ends

## **21. Game Starts / Ends**

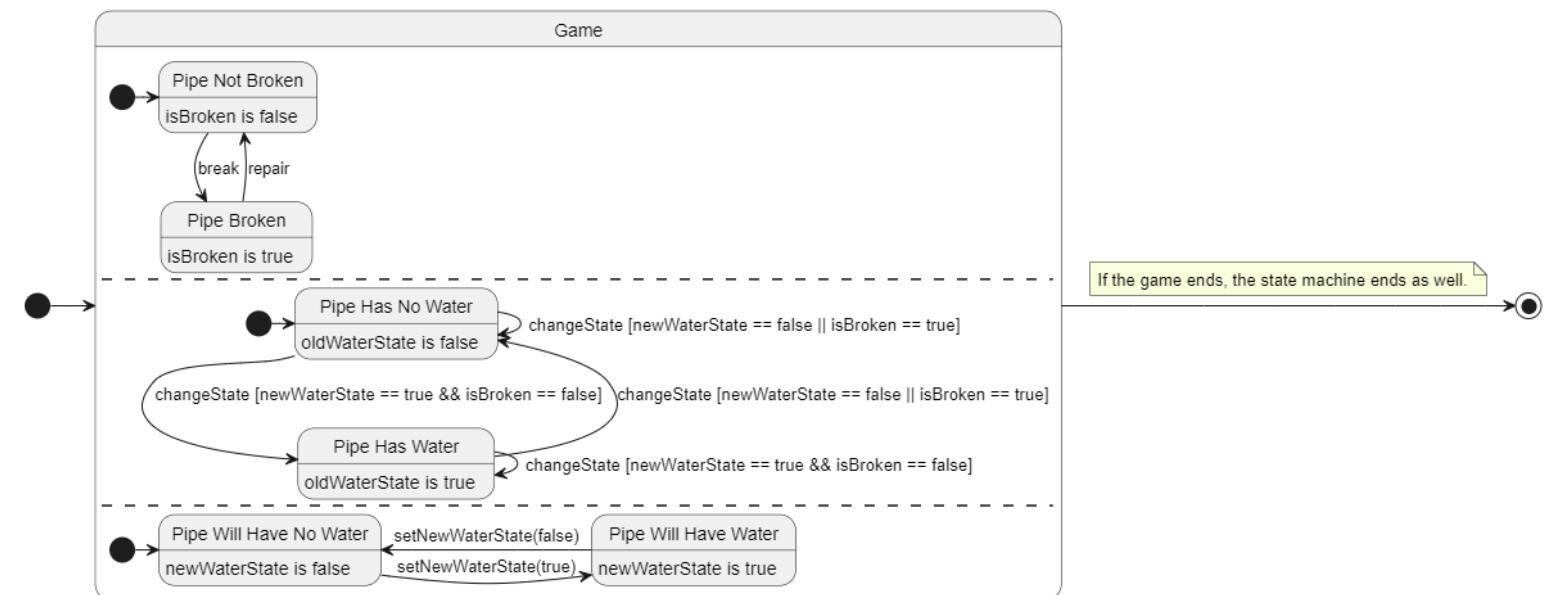


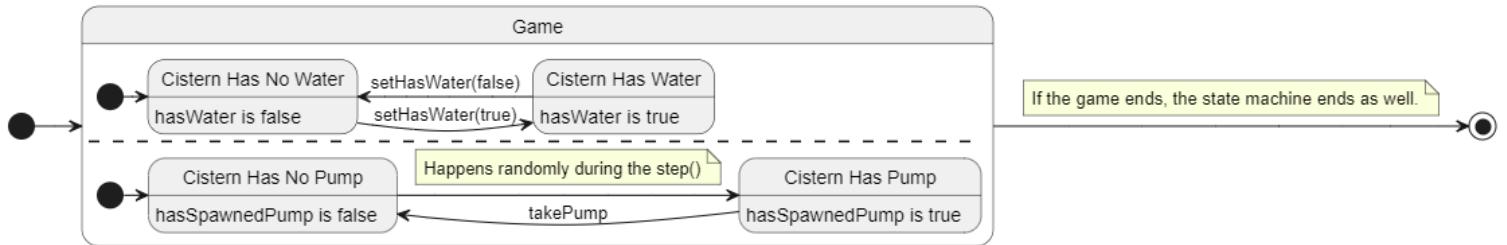
### **3.4 State-chartok**

## Pump State-chart:



## Pipe State-chart:



**Cistern State-chart:**

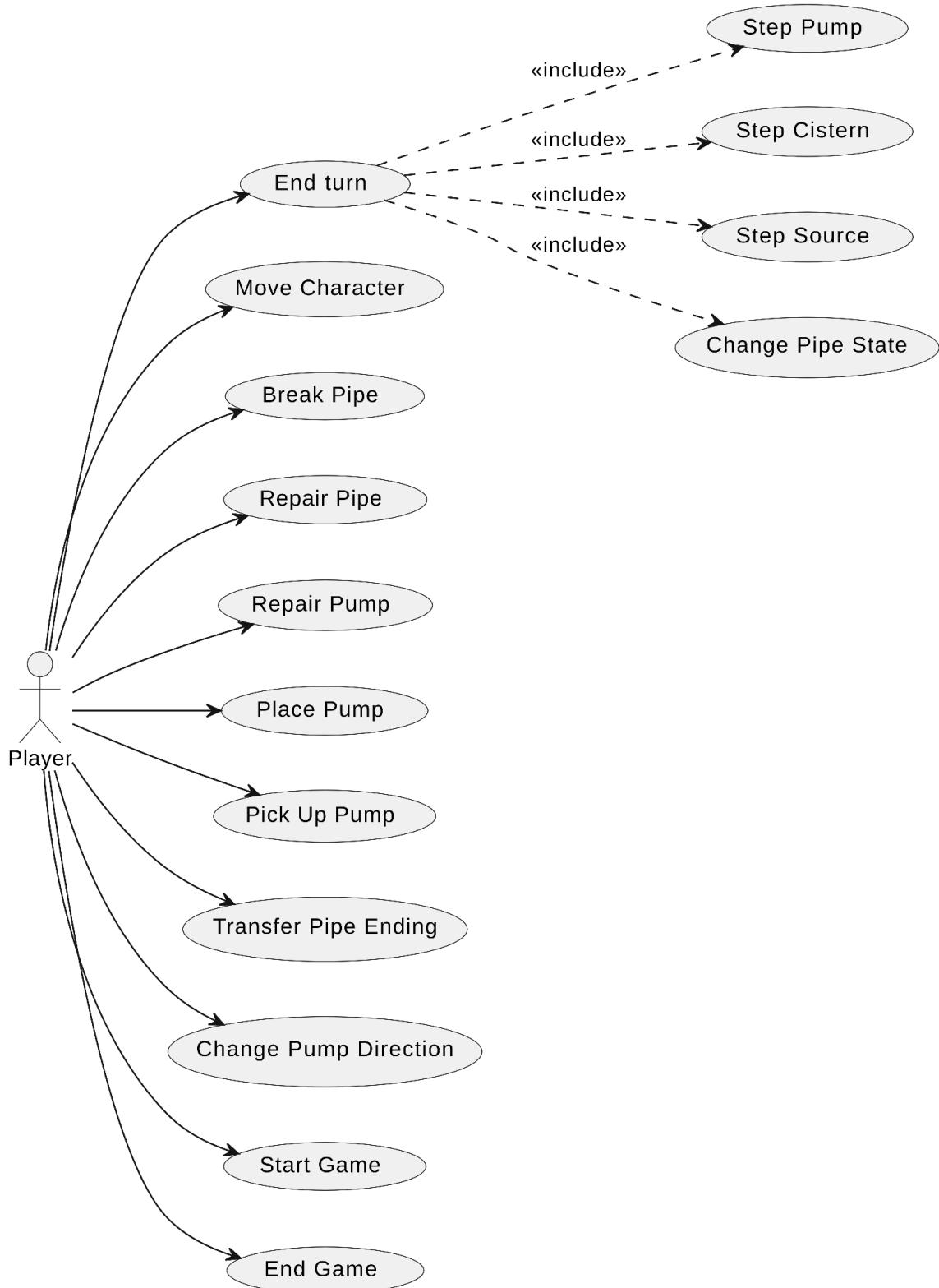
**Napló**

<b>Kezdet</b>	<b>Időtartam</b>	<b>Résznevők</b>	<b>Leírás</b>
2023.03.23. 16:00	3 óra	Garai Mali Varga Völgyesi Zavadil	Értekezlet, közös munka. Az Analízis modell I. beadás hibáinak átbeszélése, javítások megbeszélése.
2023.03.25. 19:30	1 óra	Zavadil	Osztálydiagram javítása
2023.03.25. 20:00	2 óra	Garai	State-chartok teljes javítása, osztálydiagramban hibák javítása
2023.03.25. 20:00	2 óra	Völgyesi	Szekvencia diagramok javítása
2023.03.26. 11:00	3 óra	Zavadil	Szekvenciadiagram, osztálydiagram javítások
2023.03.26. 15:00	4.5 óra	Zavadil	Szekvenciadiagram, osztálydiagram javítások, képek beillesztése a dokumentumba, formázások

## 5. Szkeleton tervezése

### 5.1 A szkeleton modell valóságos use-case-ai

#### Use-case diagram



### 5.1.1 Use-case leírások

<b>Use-case neve</b>	Break Pipe
<b>Rövid leírás</b>	A szabotör játékos egy csővezetéken állva azt kilyukasztja. Ha sikerült (eddig nem volt lyukas és most már az), akkor csökken a körben hátralévő akciók száma 1-el.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1.A.1 Az aktív játékos szabotör, ezért eltöri a csövet 1.A.2.A.1 A cső eddig nem volt lyukas, most lett az, csökken a hátralévő akciók száma 1-el. 1.A.2.B.1 A cső eddig is lyukas volt, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.B.1 Az aktív játékos szerelő, aki nem tudja kilyukasztani a csövet.

<b>Use-case neve</b>	Repair Pump
<b>Rövid leírás</b>	A Mechanic játékos a pumpán állva megjavítja, ha ez el volt romolva. Ha sikerült akkor a pumpa működőképes, és a játékos elveszt egy akciót pontot.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos azt a parancsot adja hogy javítsa meg a pumpát.
<b>Alternatív forgatókönyv</b>	1.A.1 A játékos egy szerelő.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1 A pumpa valóban el volt törve. Csökken az akciók száma 1-el. 1.A.1.A.2 A pumpa megjavul.
<b>Alternatív forgatókönyv</b>	1.A.1.B.1 A pumpa nem volt eltörve, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.B.1 A játékos egy szabotör, nem történik semmi.

<b>Use-case neve</b>	Pick Up Pump
<b>Rövid leírás</b>	A mechanic játékos egy ciszternán állva felvesz egy új pumpát, amit a leltárába eltesz.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos azt a parancsot adja ki hogy vegyen fel egy pumpát.
<b>Alternatív forgatókönyv</b>	1.A.1 A játékos egy szerelő.
	1.A.1.A.1 A játékos ciszternán áll.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.A.1 A ciszternán van pumpa, a játékos felveszi. 1.A.1.A.1.A.2 Csökken az akciók száma 1-el.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.B.1 A ciszternán nincs pumpa, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.A.1.B.1 A játékos nem ciszternán áll, nem történik semmi.

<b>Alternatív forgatókönyv</b>	1.B.1 A játékos egy szabotőr, nem történik semmi.
--------------------------------	---

<b>Use-case neve</b>	Place Pump
<b>Rövid leírás</b>	A játékos letesz egy pumpát a pályára, egy csőre illesztve
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos azt a parancsot adja ki hogy tegyen le egy pumpát a mezőre amin áll.
<b>Alternatív forgatókönyv</b>	1.A.1 A játékos egy szerelő.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1 A játékos csövön áll.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.A.1 Van a játékosnál pumpa, leteszi. 1.A.1.A.1.A.2 Csökken az akciók száma 1-el.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.B.1 Nincs a játékosnál pumpa, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.A.1.B.1 A játékos nem csövön áll, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.B.1 A játékos egy szabotőr, nem történik semmi.

<b>Use-case neve</b>	Repair Pipe
<b>Rövid leírás</b>	A Mechanic játékos megjavítja a csövet, amelyen áll. Ha valóban el volt törve, akkor a játékos veszít egy akciót pontot.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos azt a parancsot adja hogy javítsa meg a csövet.
<b>Alternatív forgatókönyv</b>	1.A.1 A játékos egy szerelő.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1 A cső valóban el volt törve. Csökken az akciók száma 1-el. 1.A.1.A.2 A cső megjavul.
<b>Alternatív forgatókönyv</b>	1.A.1.B.1 A cső nem volt eltörve, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.B.1 A játékos egy szabotőr, nem történik semmi.

<b>Use-case neve</b>	Move Character
<b>Rövid leírás</b>	Az aktív játékos a saját mezőjéről egy másik mezőre lép.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1.A.1.A.1.A.1 A szabotőr/szerelő játékos egy szomszédos csőre próbál lépni, amin más játékos nem áll (tehát a lépés sikeres). Először kiveszi magát a pumpa/ciszterna/forrás játékosainak listájából amin éppen áll. 1.A.1.A.1.A.2 Csökkenti a körben hátralévő akciók számát. 1.A.1.A.1.A.3 Hozzáadja magát a megcélzott cső szomszédjainak listájához (ezzel rááll).

<b>Alternatív forgatókönyv</b>	1.A.1.A.1.B.1 A megcélzott csövön áll más játékos, ezért nem léphet ide.
<b>Alternatív forgatókönyv</b>	1.A.1.B.1.A.1 A megcélzott cső nem szomszédos azzal a mezővel, amin éppen áll a játékos, ezért nem léphet ide.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.A.1 A szabotőr/szerelő játékos egy szomszédos pumpára/ciszternára/forrásra próbál lépni (ha ez teljesül, a lépés sikeres). Először kiveszi magát a cső játékosainak listájából amin éppen áll. 1.A.1.A.1.A.2 Csökkenti a körben hátralévő akciók számát. 1.A.1.A.1.A.3 Hozzáadja magát a megcélzott pumpa/ciszterna/forrás szomszédjainak listájához (ezzel rááll).
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.B.1 A megcélzott pumpa/ciszterna/forrás nem szomszédos azzal a csővel, amin a játékos éppen áll, ezért nem léphet ide.

<b>Use-case neve</b>	Transfer Pipe Ending
<b>Rövid leírás</b>	A játékos egy kijelölt csőnek a csatlakozási pontját áthelyezi egy másik mező szabad bemenetére.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos kiadja a parancsot, amellyel áthelyezhet egy csővéget.
<b>Alternatív forgatókönyv</b>	1.A.1 Ugyanazt az elemet adja meg kimenetnek, mint ami a bemenet, ezért nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.B.1 Érvényes helyre köti a csövet. 1.B.2 A cső régi csatlakozási pontjának szomszédai közül töröljük a csövet. 1.B.3 A cső szomszédai közül töröljük a régi csatlakozási pontját. 1.B.4 A cső szomszédaihoz hozzáadjuk az új csatlakozási pontját. 1.B.5 Az új csatlakozási pont szomszédaihoz hozzáadjuk a csövet, eggyel csökken a játékos akciójának száma.

<b>Use-case neve</b>	Change Pump Direction
<b>Rövid leírás</b>	A játékos a pumpán állva, átállítja az aktív bemenetet vagy kimenetet.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos kiadja az irány átállítására szolgáló parancsot.
<b>Alternatív forgatókönyv</b>	1.A.1 A játékos pumpán áll.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1 A játékos érvényes (a pumpával szomszédos) be- és kimeneteket ad meg.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.A.1 A megadott be- és kimenet nem ugyanaz a pályaelem, megtörténik az átállítás, az akciók száma 1-el csökken.
<b>Alternatív forgatókönyv</b>	1.A.1.A.1.B.1 A megadott be- és kimenet ugyanaz a pályaelem, nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.A.1.B.1 A megadott be- vagy kimenet érvénytelen (nem szomszédos a pumpával), nem történik semmi.

<b>Alternatív forgatókönyv</b>	1.B.1 A játékos nem pumpán áll, nem történik semmi.
--------------------------------	---

<b>Use-case neve</b>	Game Starts / Ends
<b>Rövid leírás</b>	A játék indításakor léterjön legalább két szerelő és legalább két szabotőr karakter, 3-3 akcióponttal inicializálva.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	<p>1. A játékos elindítja a játékot.</p> <p>2. A játékbeli események addig zajlanak, amíg egy csapat el nem éri a győzelemhez szükséges pontszámot.</p> <p>3. A pontszám elérése után leáll a játék.</p>

<b>Use-case neve</b>	Step Pump
<b>Rövid leírás</b>	A pumpa köre végén elromolhat. Ha nincs elromolva akkor a tartályból a kimenetre helyez vizet. Ha a bemeneten van víz akkor a tartály vizét innen pótolja.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1.A.1 A pumpa elromlik a kör végén (vagy már elromlott korábban). Ebben a körben nem pumpál vizet.
<b>Alternatív forgatókönyv</b>	1.B.1.A.1 A pumpa nem romlott még el, tartályában van víz, ilyenkor a pumpa a kimeneti csövére tesz vizet.
<b>Alternatív forgatókönyv</b>	1.B.1.B.1 A pumpa nem romlott még el, tartályában nincs víz, ilyenkor a kimeneti csövről elveszi a vizet.
<b>Alternatív forgatókönyv</b>	1.B.2.A.1 A pumpa nem romlott még el, bemeneti csövén van víz, ilyenkor a tartályában lesz víz.
<b>Alternatív forgatókönyv</b>	1.B.2.B.1 A pumpa nem romlott még el, bemeneti csövén nincs víz, ilyenkor a tartályában nem lesz víz.

<b>Use-case neve</b>	Step Cistern
<b>Rövid leírás</b>	A ciszterna véletlenszerűen egy pumpát generál, majd utána a pontszám növelésével foglalkozik.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1.A.1 A ciszterna generál pumpát az adott körben.
<b>Forgatókönyv</b>	1.B.1 A ciszterna nem generál pumpát az adott körben.
<b>Alternatív forgatókönyv</b>	2.A.1. A ciszterna bemeneti csövén van víz (oldWaterState), a pontot kapnak a szerelők.
<b>Alternatív forgatókönyv</b>	2.B.1. A ciszterna bemeneti csövén nincs víz, semmi nem történik.

<b>Use-case neve</b>	Step Source
----------------------	-------------

<b>Rövid leírás</b>	A forrásra csatlakoztatott cső minden kap vizet.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1.A.1 A forrásnak van kimeneti csöve, erre tesz vizet a forrás.
<b>Alternatív forgatókönyv</b>	1.B.1 A forrásra nincs csatlakoztatva cső, nem történik semmi.

<b>Use-case neve</b>	Change Pipe State
<b>Rövid leírás</b>	A csőben amennyiben van víz és lyukas, növeli a szabotörök pontszámát
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1.A.1 A cső nincs eltörve vagy nincs benne víz (oldWaterState), ilyenkor nem történik semmi.
<b>Alternatív forgatókönyv</b>	1.B.1 A cső el van törve és van benne víz (oldWaterState), a szabotörök kapnak pontot.

<b>Use-case neve</b>	End turn
<b>Rövid leírás</b>	A játékos befejezi a körét. Meghívódnak az egyes komponenseknek a kör végi függvényei (step() és changeState()), ezeket külön use case-eken és hozzájuk tartozó diagramokon ábrázoljuk: Step Pump Step Cistern Step Source Step Pipe Change Pipe State
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	1. A játékos befejezi a körét. A különböző komponensek (cső, ciszterna, forrás, pumpa) elvégzik a kör végi teendőiket.

## 5.2 A szkeleton kezelői felületének terve, dialógusok

Jelölések:

- “a” - “a” nevű példány
- “[ A ]” - “A” osztály
- “->” - függvényhívás
- “ab()” - “ab” nevű függvény

Műveletek, kifejezések:

- [ A ]a - az “a” nevű objektum az “A” osztály példánya
- [ Actor ]a - az “a” nevű objektum egy aktor
- a -> b - az “a” nevű objektum meghív egy függvényt a “b” nevű objektumon
- a -> b: ab() - az “a” nevű objektum meghívja az “ab” függvényt a “b” nevű objektumon
- ab(e, f, ...) - az “ab” függvény argumentumai az {"e", "f", ...} példányok

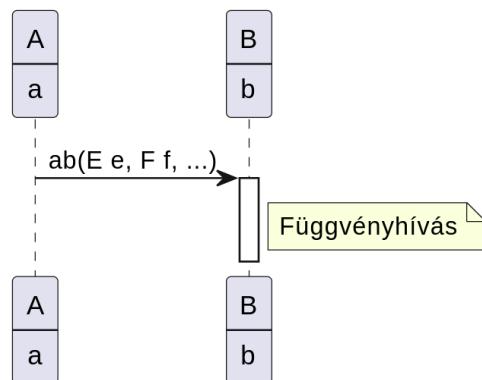
- **ab()**: [M] - az "ab" függvény visszatérési értéke az "M" osztály egy példányá, vagy "M" primitív típusú
  - **return a** - a legutolsóként meghívott függvény visszatér az "a" objektummal
  - **return** - a legutolsóként meghívott függvény visszatér érték nélkül

### A szkeleton által adott kimenetek:

"A" osztály "a" példánya meghívja a "B" osztály "b" példányát az "ab" függvényel, melynek argumentumai: {"E" osztály "e" példánya, "F" osztály "f" példánya, ...}. A függvény visszatérési értéke az "M" osztály egy példánya vagy "M" típusú. Ez a következőképp fog megjelenni a képernyőn:

[A]a -> [B]b: ab([E]e, [F]f, ...): [M]

*Ez a viselkedés a szekvenciadiagramon:*



*Amennyiben egy lifeline-hoz tartozó objektum többféle osztálynak is lehet a példánya, azt a következőképp jelöljük:*

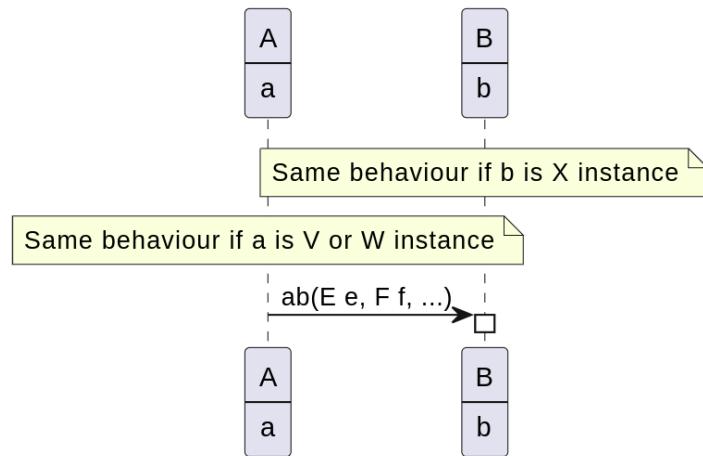
[A/V/W/...]a

Ez egy függvényhívásnál például a következőképp néz ki:

[A/V/W]a -> [B/X]b: ab([E]e, [F]f, ...): [M]

Itt tehát az "a" példány lehet az "A" vagy "V" vagy "W" osztály példánya, a "b" lehet a "B" vagy "X" osztályok példánya. Az "e" és "f" osztályok továbbra is csak rendre az "E" és "F" osztály példányai lehetnek. Ilyen jelölést akkor használunk, ha a viselkedés szempontjából több osztálynak a példányai is **pontosan ugyanazt** csinálják.

Ez a viselkedés szekvenciadiagramon kommentekkel van jelölve a következő módon:

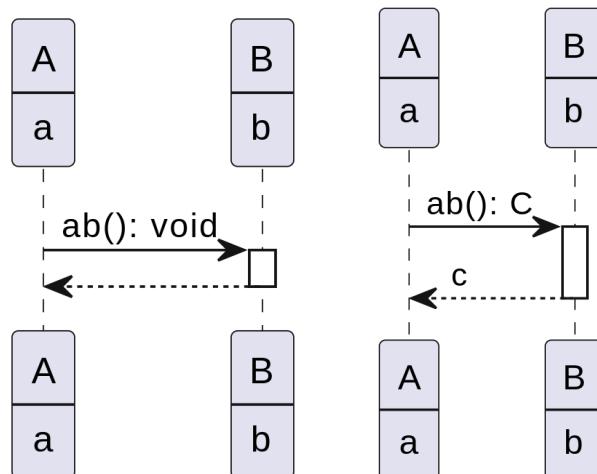


*Amennyiben egy függvénynek nincs visszatérési értéke, ezt a `void` szóval jelöljük:*

```
[A]a -> [B]b: ab([E]e, [F]f, ...): void
```

*Egy függvény, amikor visszatér, a `return` szóval jelöljük. Példa függvényhívásra és visszatérésre:*

```
[A]a -> [B]b: ab(): void
return
[A]a -> [B]b: ab(): [C]
return c
```

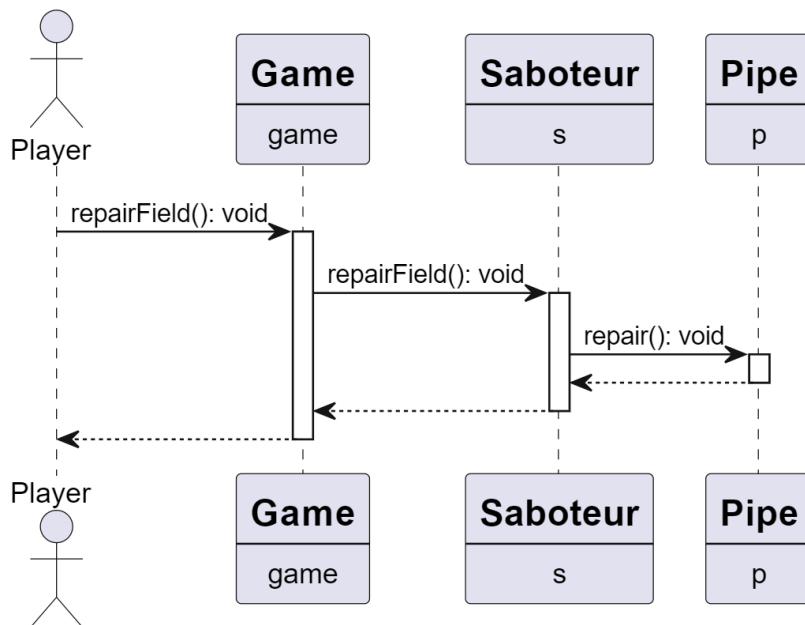


*Példa szkeleton kimenet:*

```
[Actor]Player -> [Game]game: repairField(): void
[Game]game -> [Saboteur]s: repairField(): void
    [Saboteur]s -> [Pipe]p: repair(): void
        return
    return
return
```

*A következő szekvencia tartozik hozzá:*

### 8.B Repair Pipe



A szkeleton program a fent látható indentálással könnyíti majd meg az átláthatóságot.

#### A szkeleton program bemenetei:

A szkeleton program indulás után kilistázza a felhasználónak az összes lehetséges use case-t, amelyeket le lehet futtatni benne. Mindegyik előtt szerepel egy sorszám, a felhasználó az egyik ilyet beírva elindíthatja az adott szekvenciának megfelelő futást.

Egy use case kiválasztása után a program adott esetben megkérdezheti a felhasználót, hogy melyik forgatókönyv játszódjon le (pl. Move Character use case-en belül szabotör / szerelő lépjen, ciszternára vagy csőre stb.). Ezeket az alternatív forgatókönyveket kilistázza egymás alá, mindegyik előtt szerepel egy betű (A,B,C, ...). A felhasználó ezek közül a megfelelő betű beírásával tud választani.

A szkeleton a futása közben, minden függvényhívás után, ha az adott függvénynek van paramétere, melyet a felhasználónak kell megadnia, megkérdezi a felhasználót, hogy milyen argumentumot szeretne megadni. Ha több paramétere van, mindegyikkel ezt teszi. Például egy kétparaméteres függvény esetén:

Szkeleton kimenete:

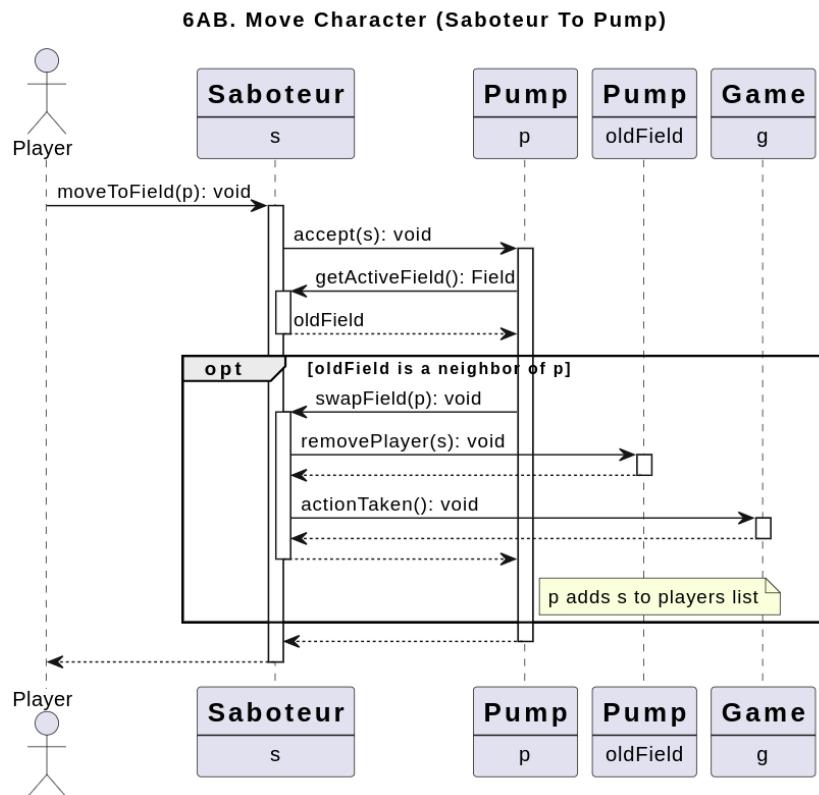
```

[A]a -> [B]b: ab([E] e, [F] f): void
e: eInstance
f: fInstance
...
return
  
```

A zöld színű és aláhúzott szavak a felhasználó által beírt (a szekvenciában létező) objektumok nevei.

Amennyiben a szekvenciadiagramon egy ōrfeltétel van (pl. opt, alt stb), az ōrfeltételt teljes egészében kiírja a szkeleton program, erre a felhasználónak egy True vagy False értéket kell beírnia. Ennek megfelelően fognak az egyes események lefutni.

Példa ōrfeltételekre:



A fenti szekvenciadiagramhoz tartozó szkeleton kód egy részlete (zölddel és aláhúzással vannak jelölve a felhasználói bemenetek):

```

...
return oldField
[oldField is a neighbor of p]: True
[Pump]p -> [Saboteur]s: swapField(p): void
    [Saboteur]s -> [Pump]oldField: removePlayer(s): void
...

```

Amennyiben a felhasználó **False**-t ír be bemenatként a fenti szekvencia a következőképp alakul:

```

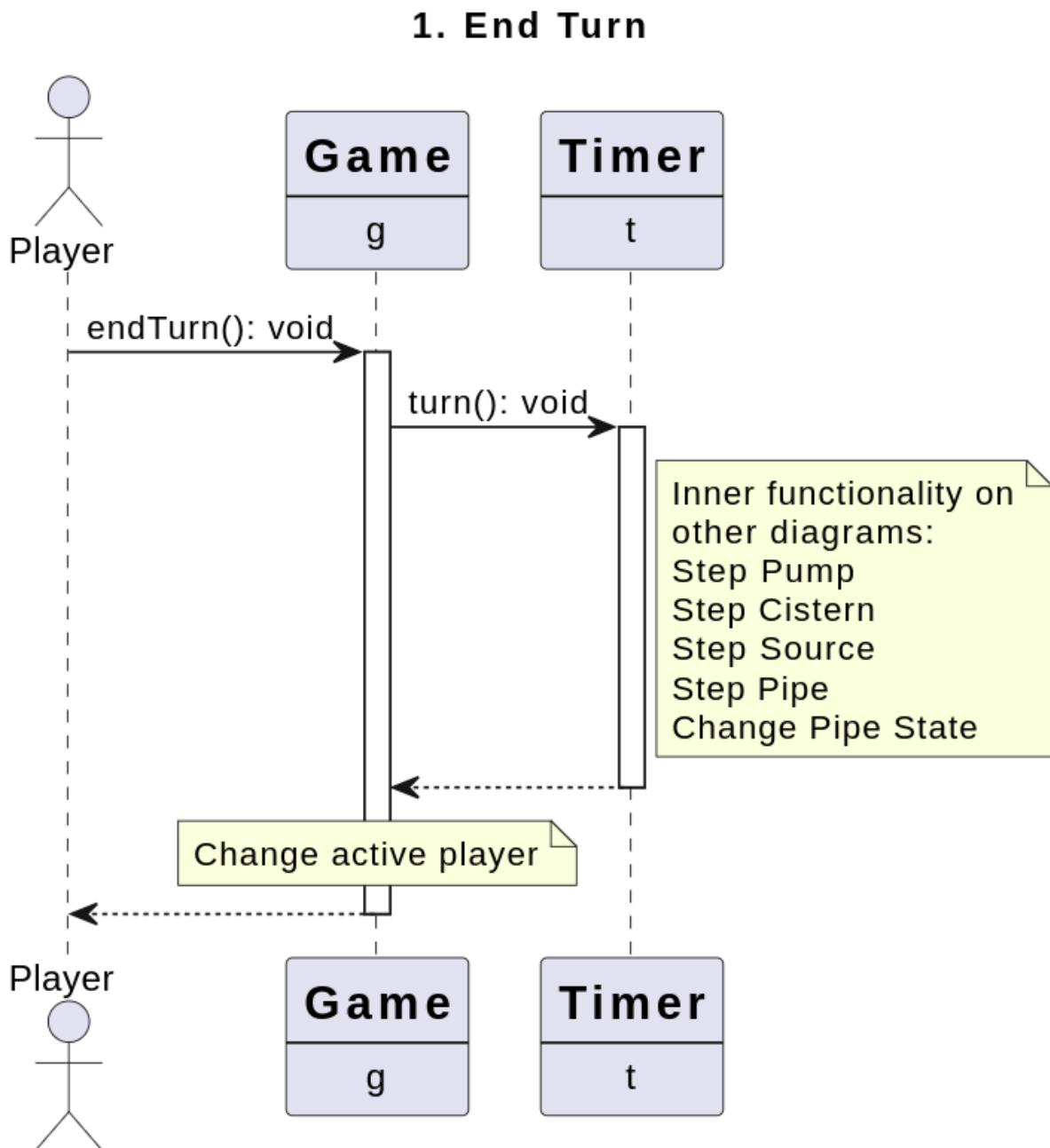
...
return oldField
[oldField is a neighbor of p]: False
return
return
...

```

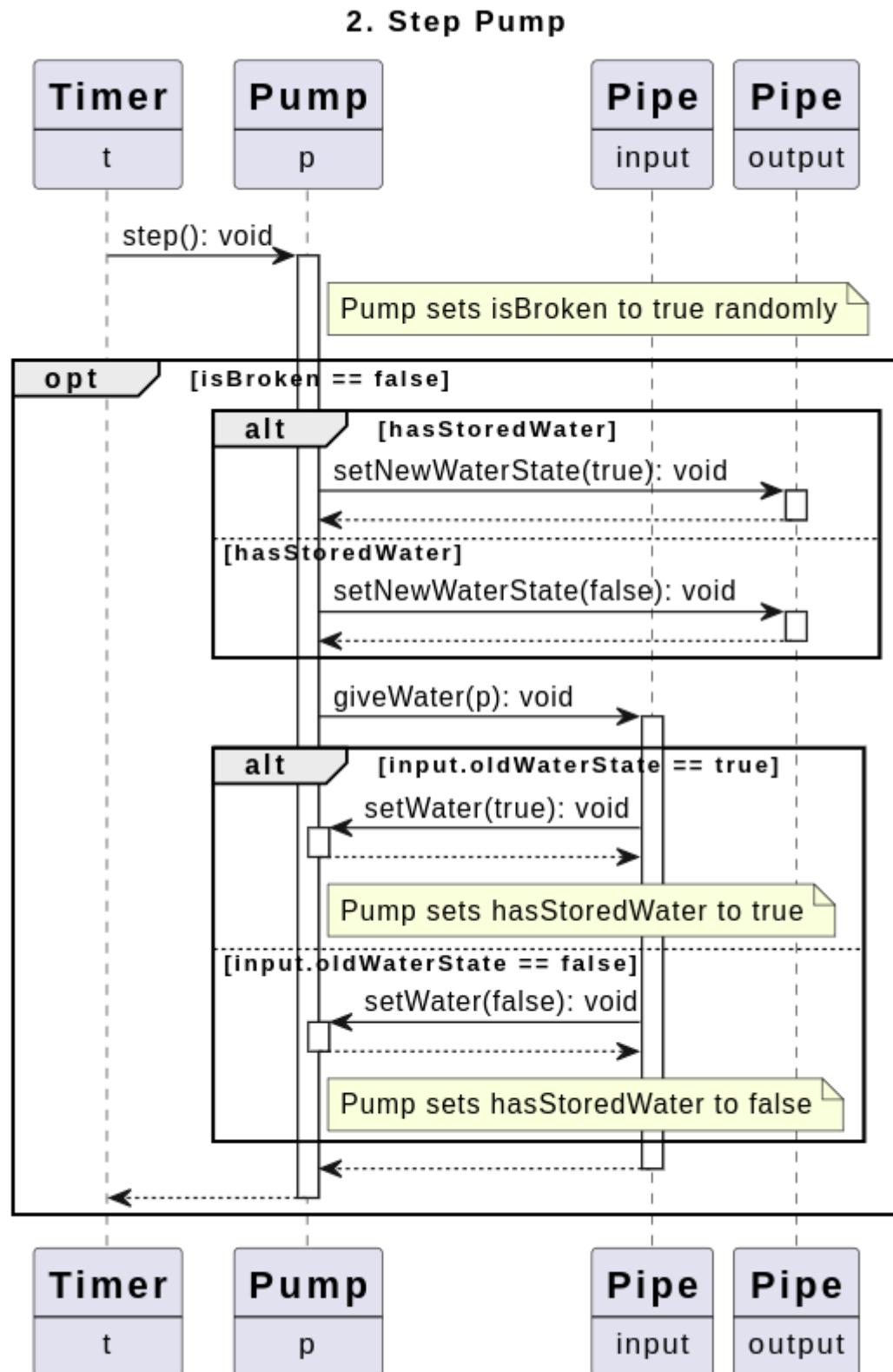
### 5.3 Szekvencia diagramok a belső működésre

*Megjegyzés: A Game és a Timer statikus osztályokat singleton osztályokként modellezzük (azért, hogy a szkeleton programban példányosítani tudjuk őket).*

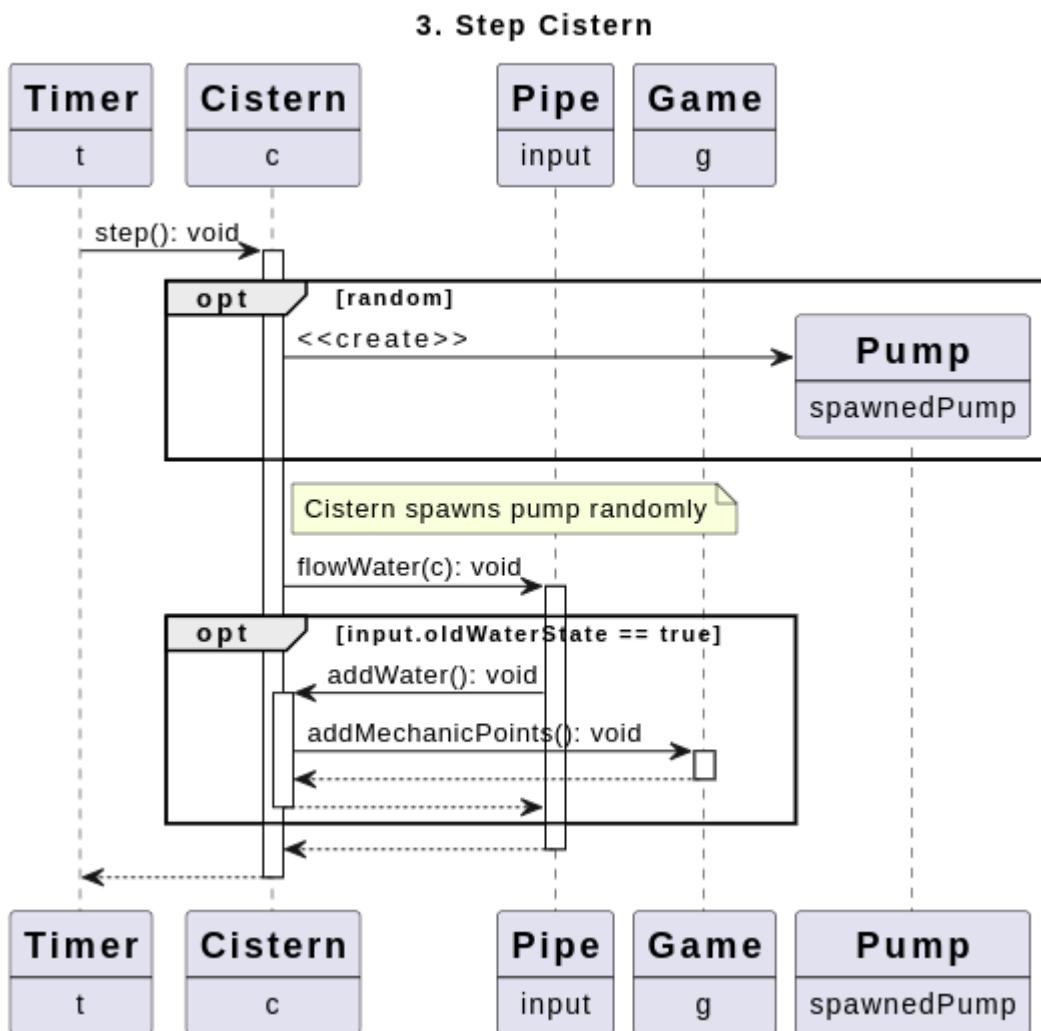
1. End Turn



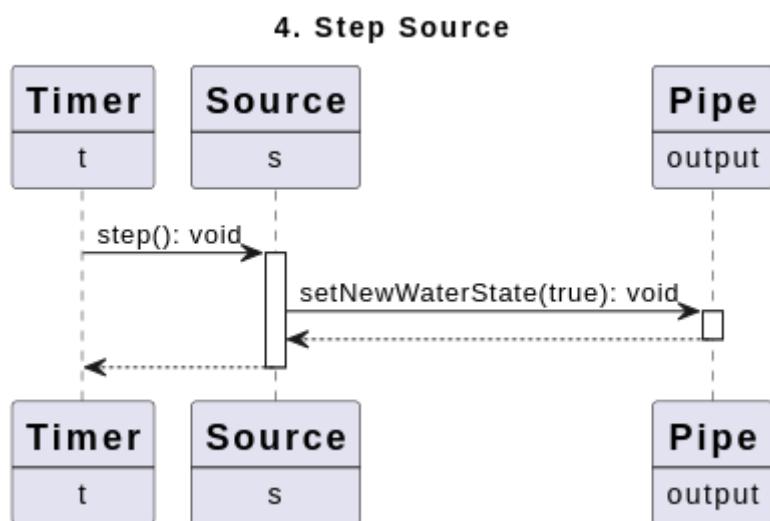
## 2. Step Pump



## 3. Step Cistern

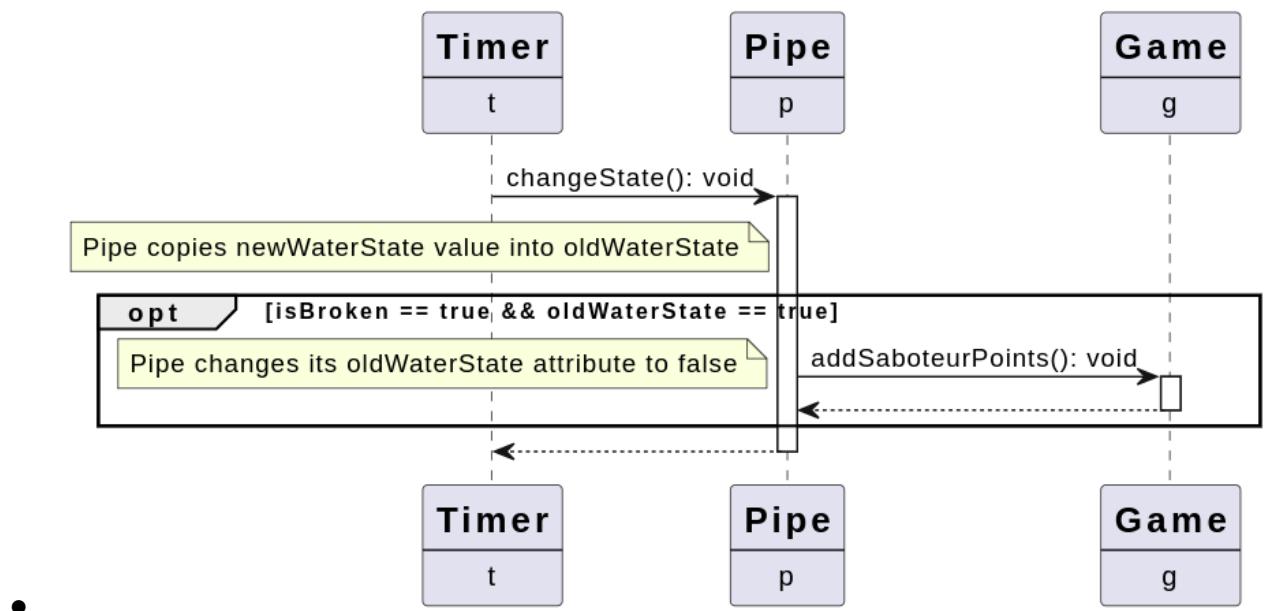


## 4. Step Source

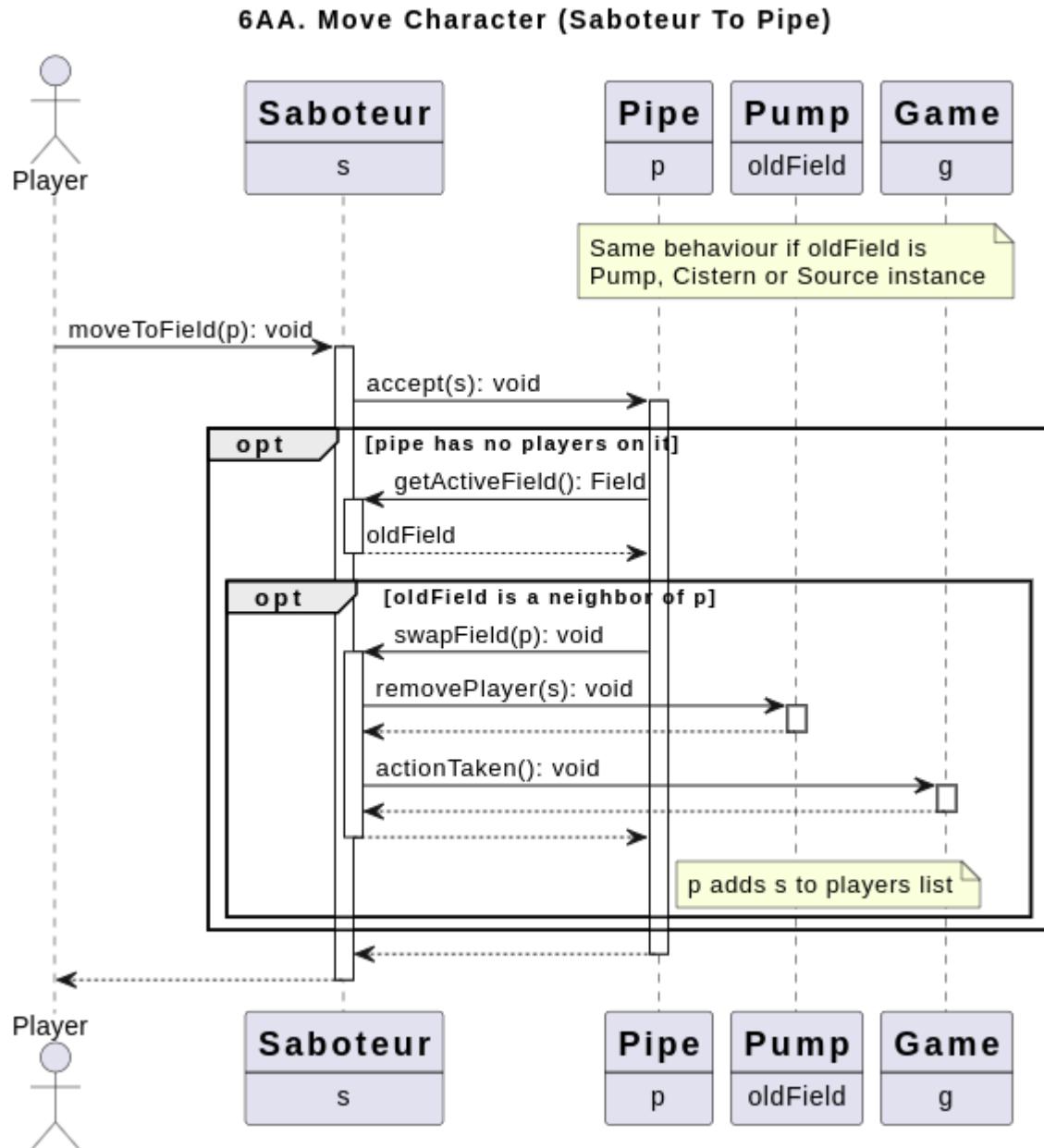


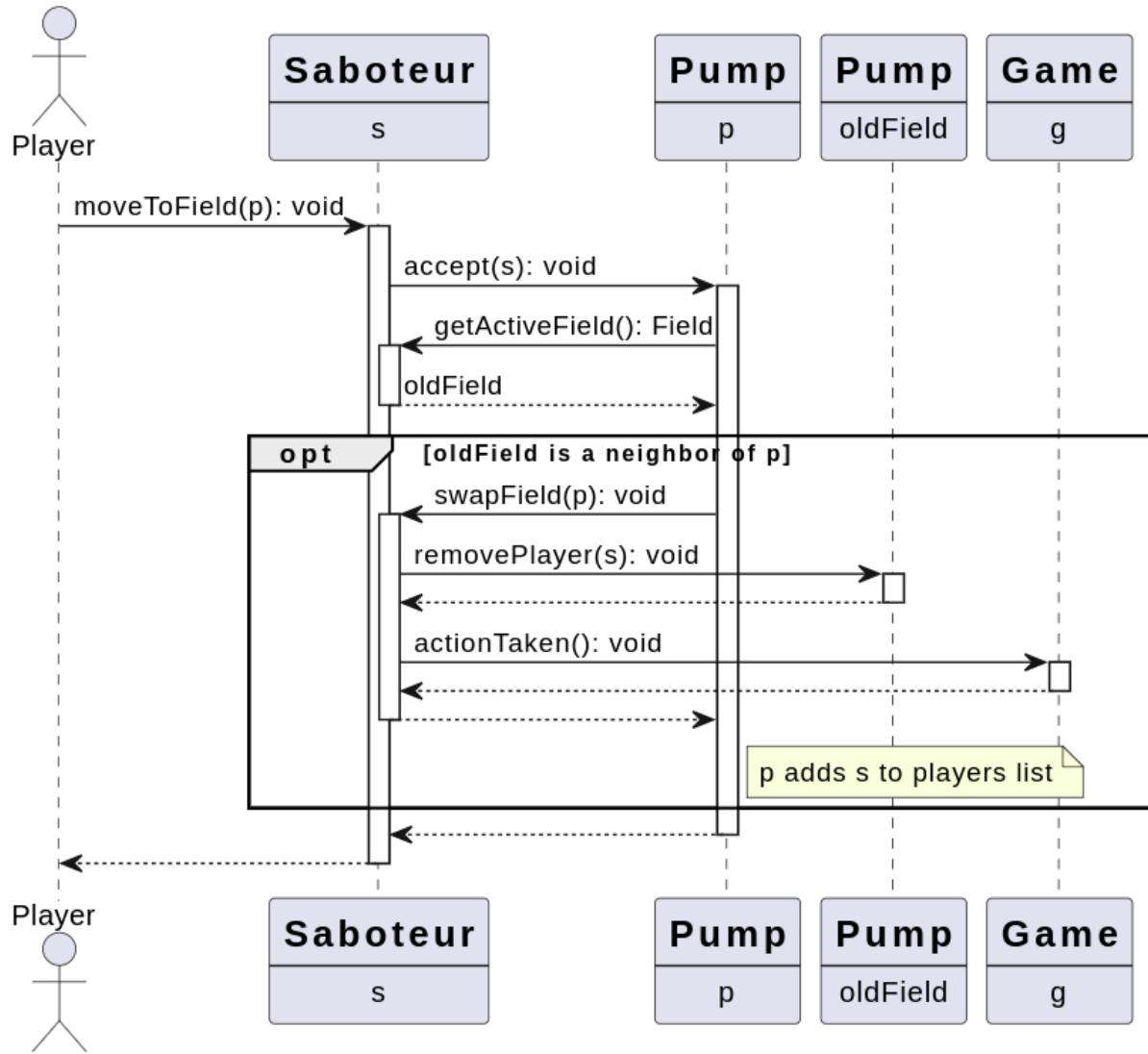
## 5. Change Pipe State

## 5. Change Pipe State

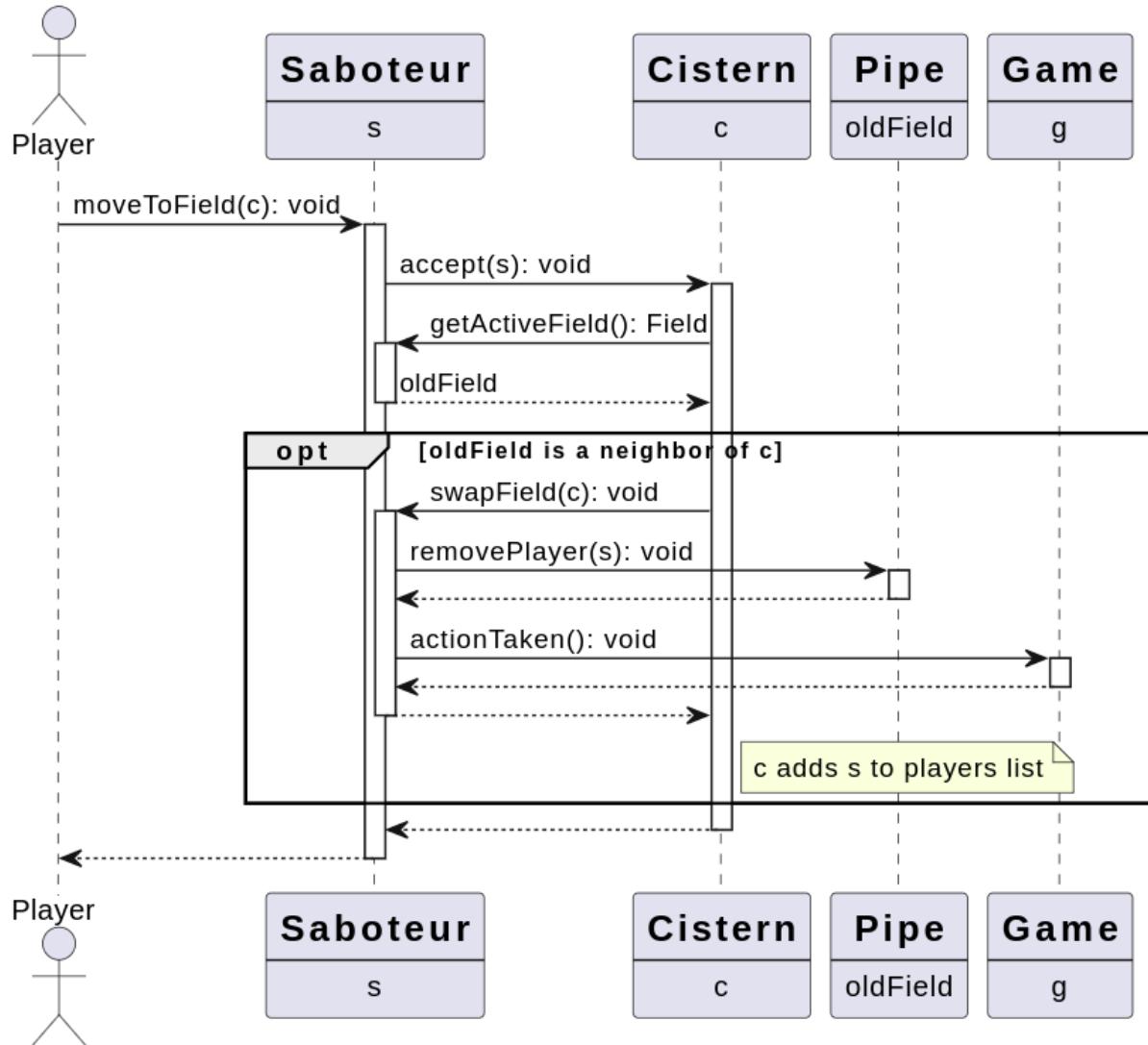


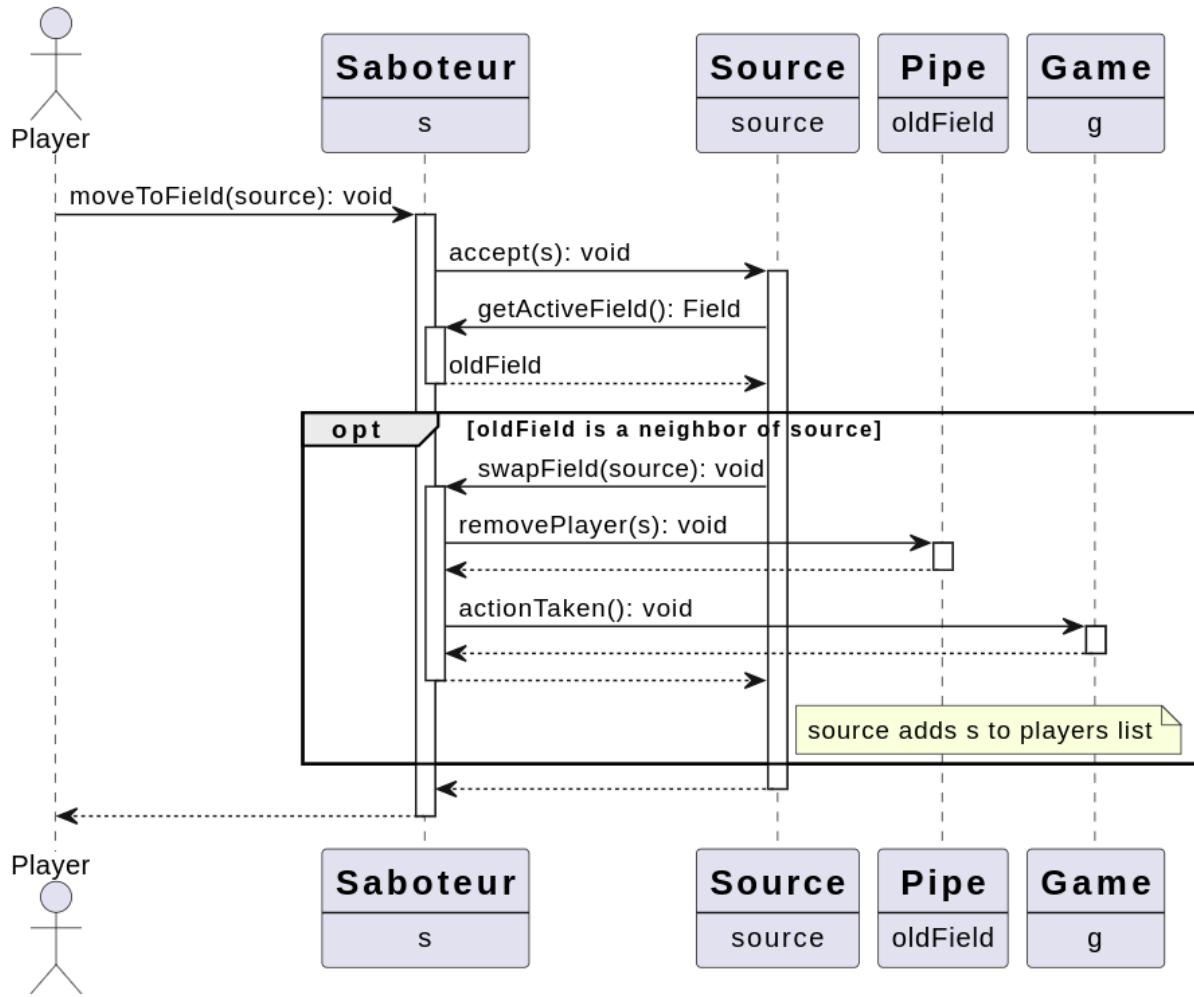
6. Move Character  
 a. Saboteur To Pipe



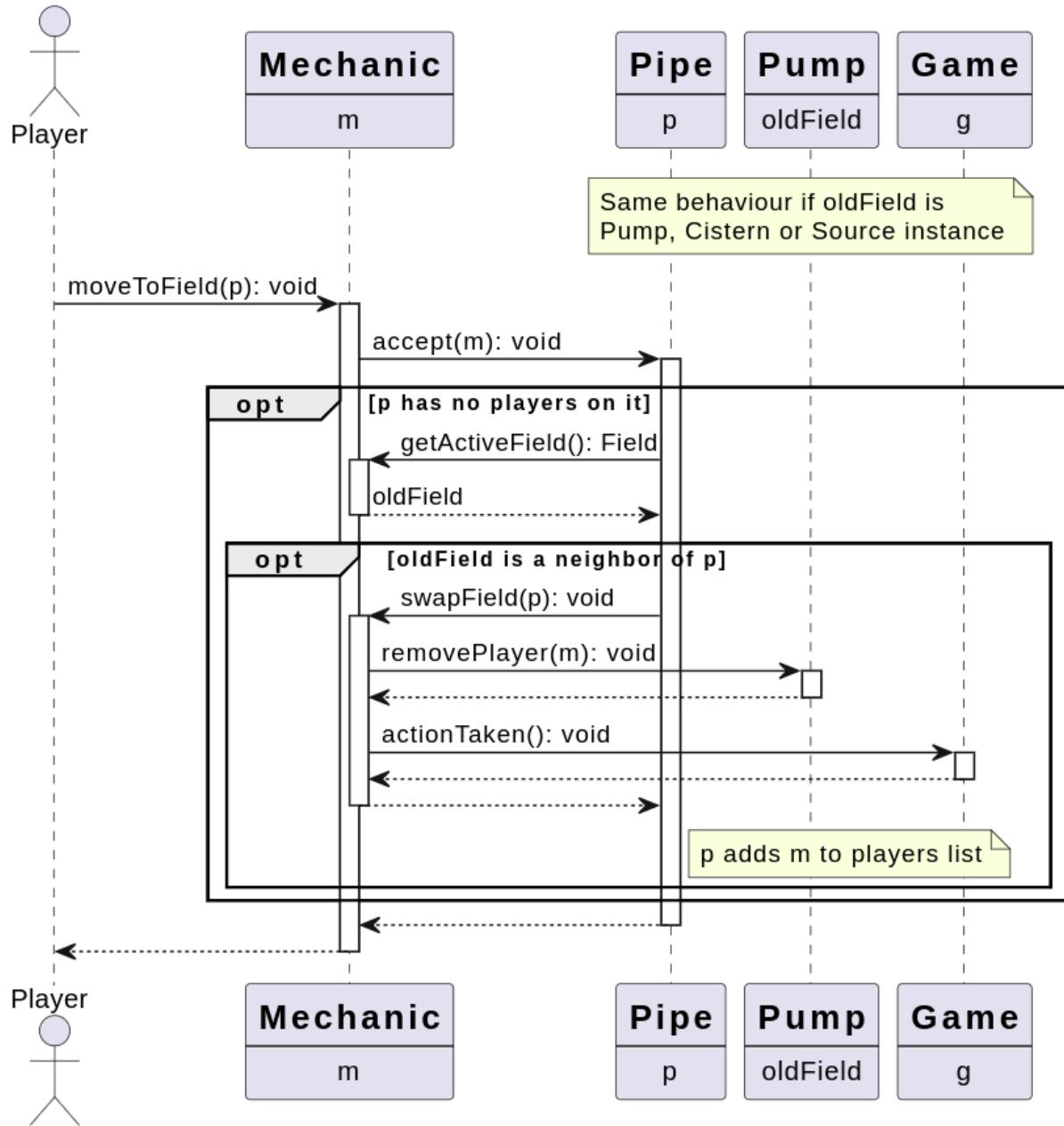
*b. Saboteur To Pump***6AB. Move Character (Saboteur To Pump)**

## c. Saboteur To Cistern

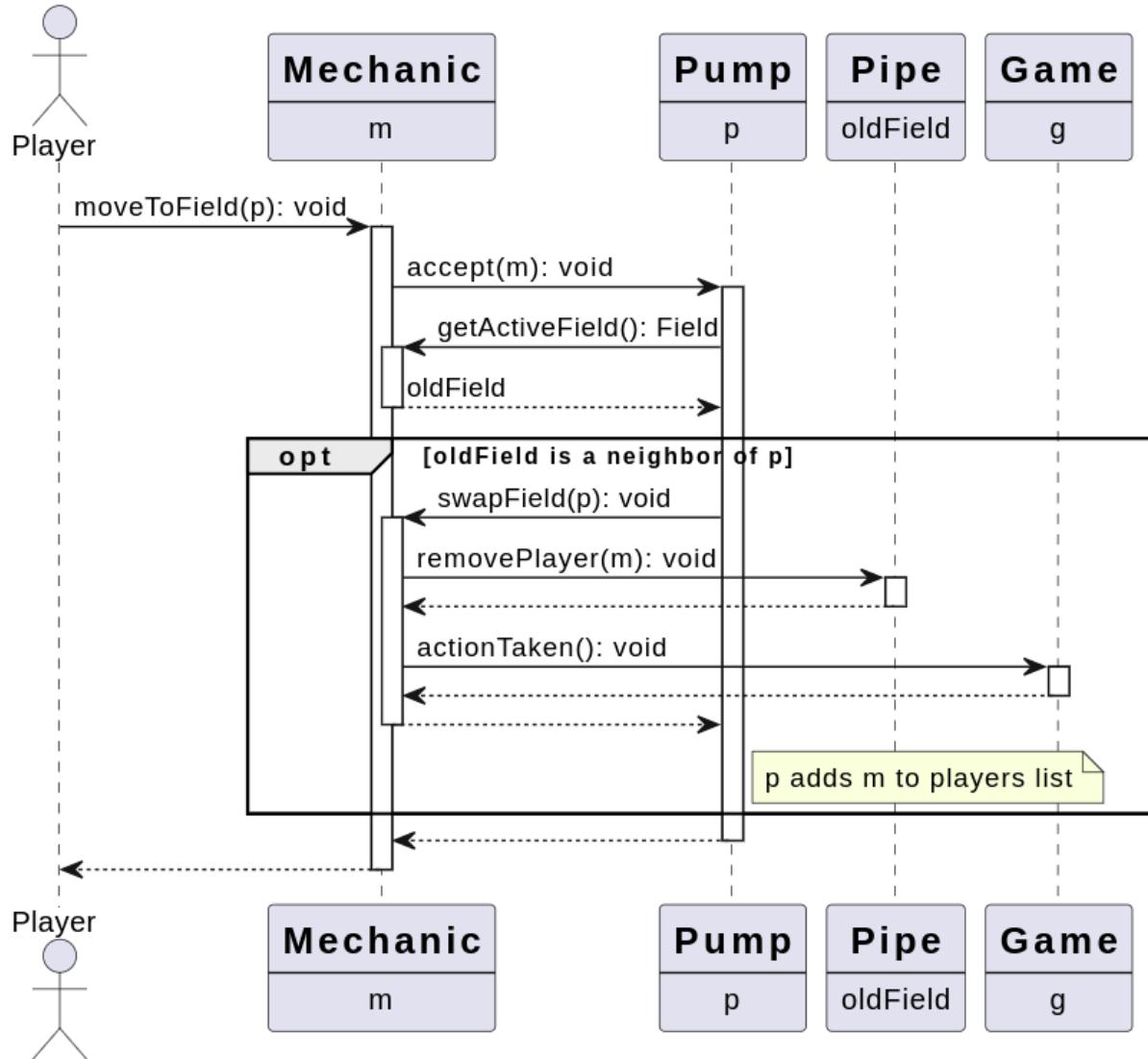
**6AC. Move Character (Saboteur To Cistern)**

*d. Saboteur To Source***6AD. Move Character (Saboteur to Source)**

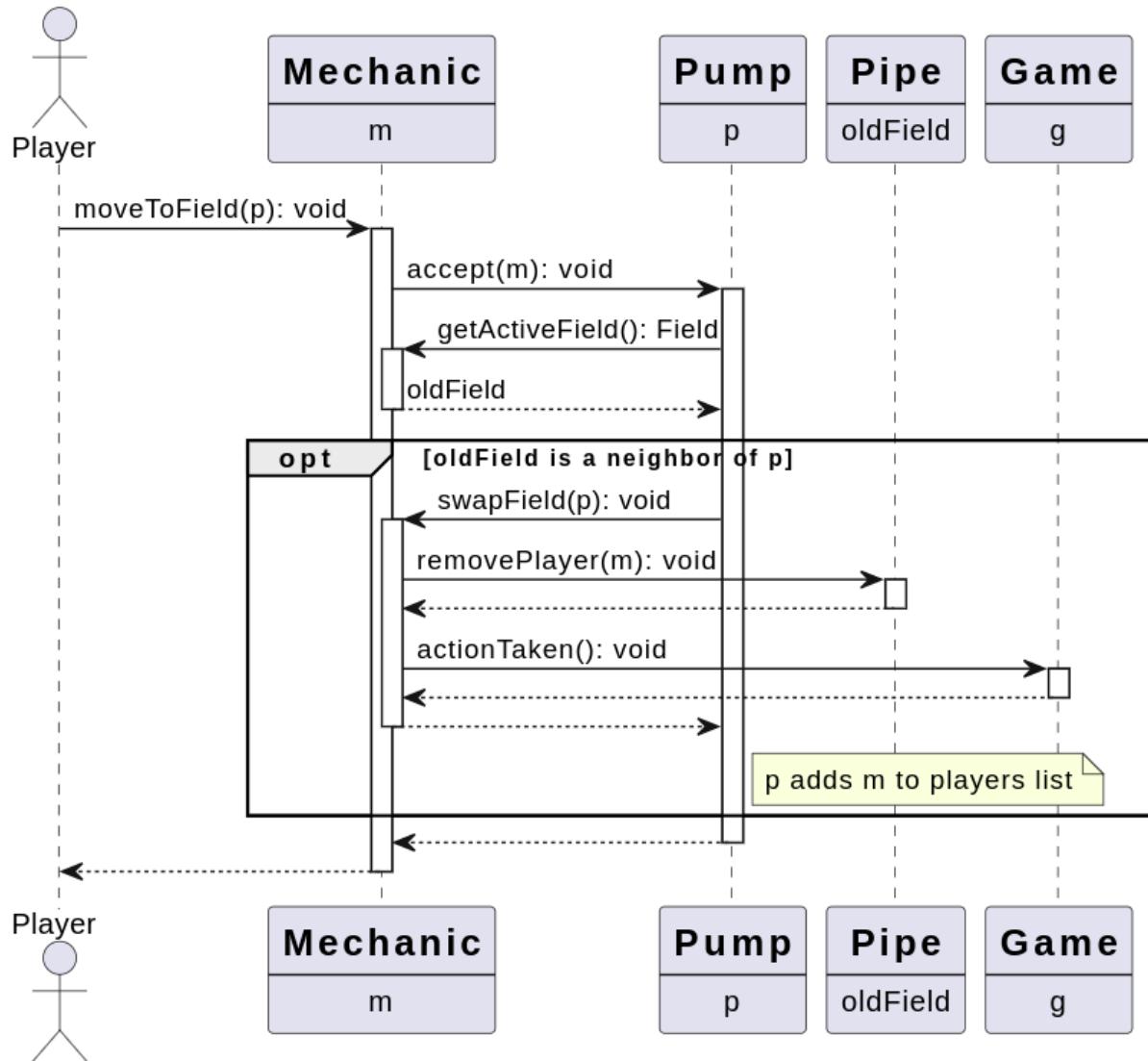
## e. Mechanic To Pipe

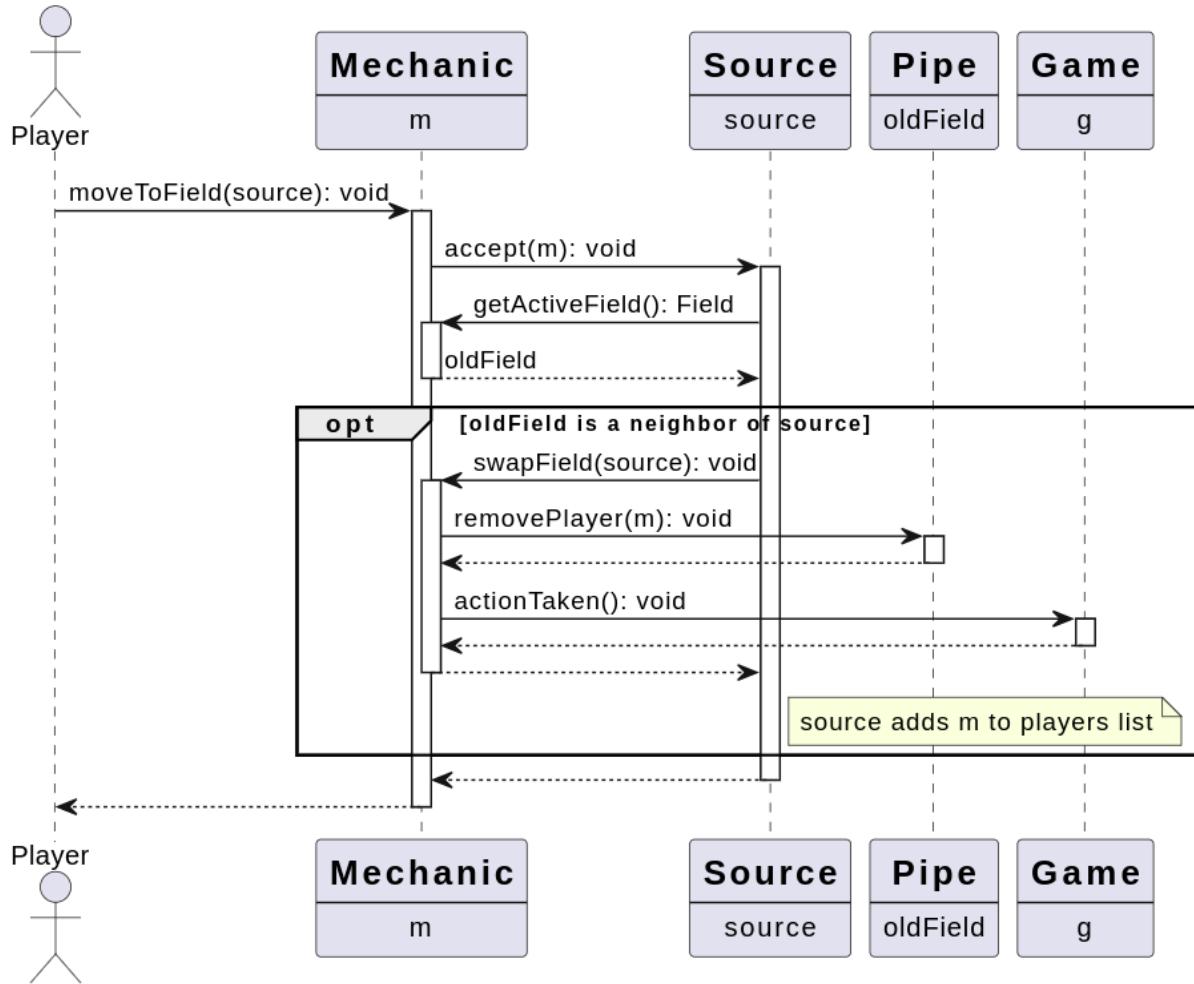
**6BA. Move Character (Mechanic To Pipe)**

## f. Mechanic To Pump

**6BB. Move Character (Mechanic To Pump)**

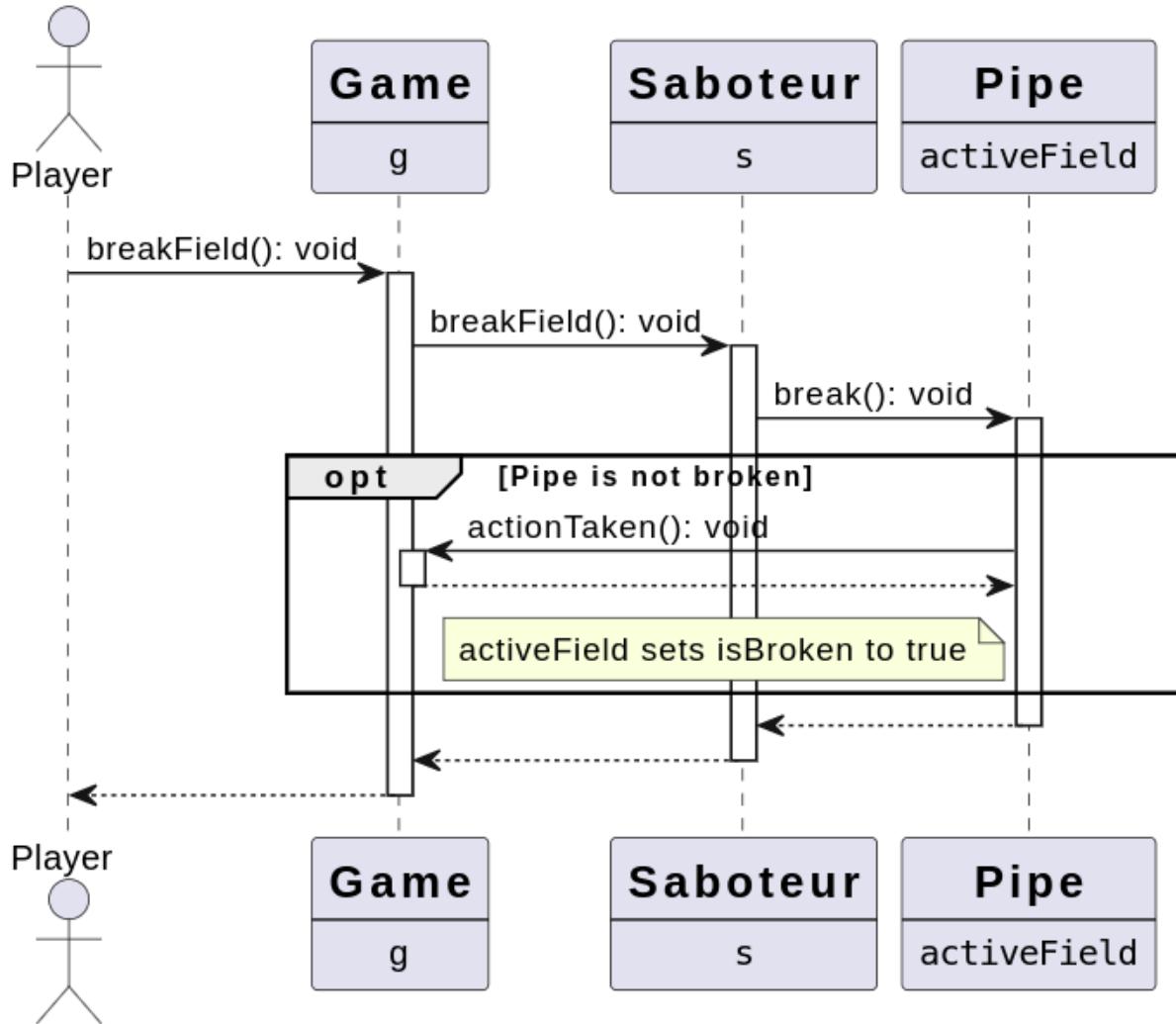
## g. Mechanic To Cistern

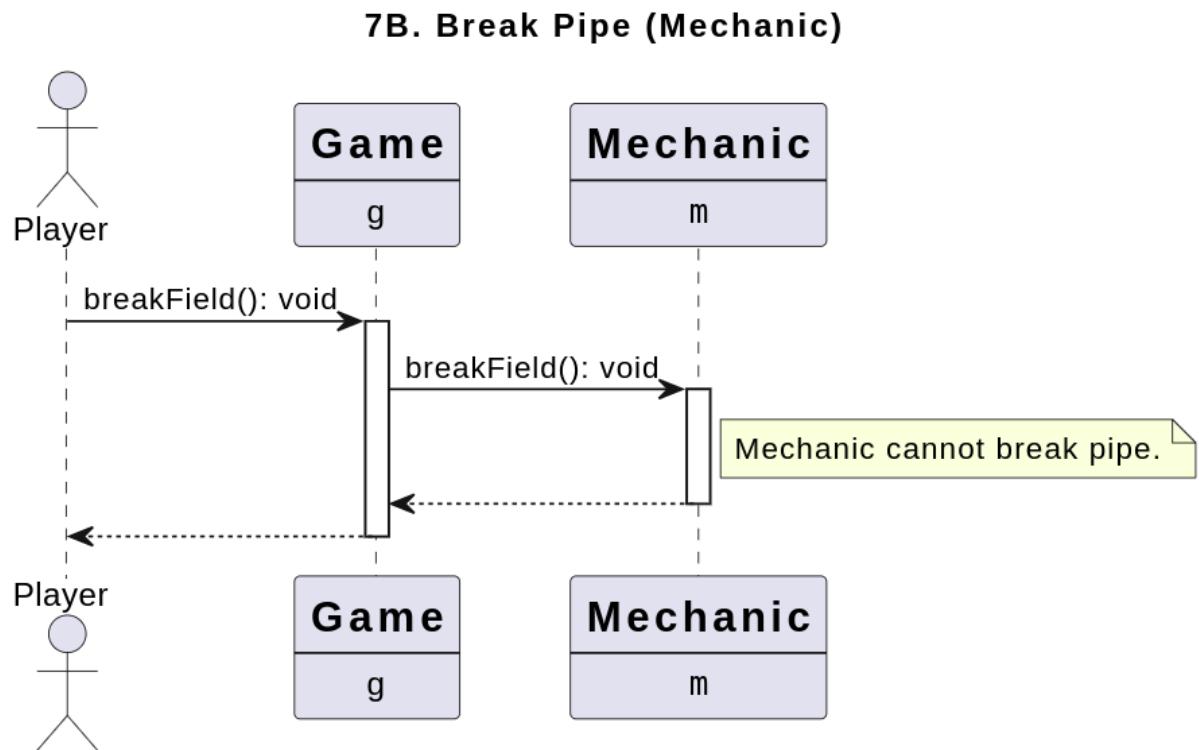
**6BB. Move Character (Mechanic To Pump)**

*h. Mechanic To Source***6BD. Move Character (Mechanic To Source)**

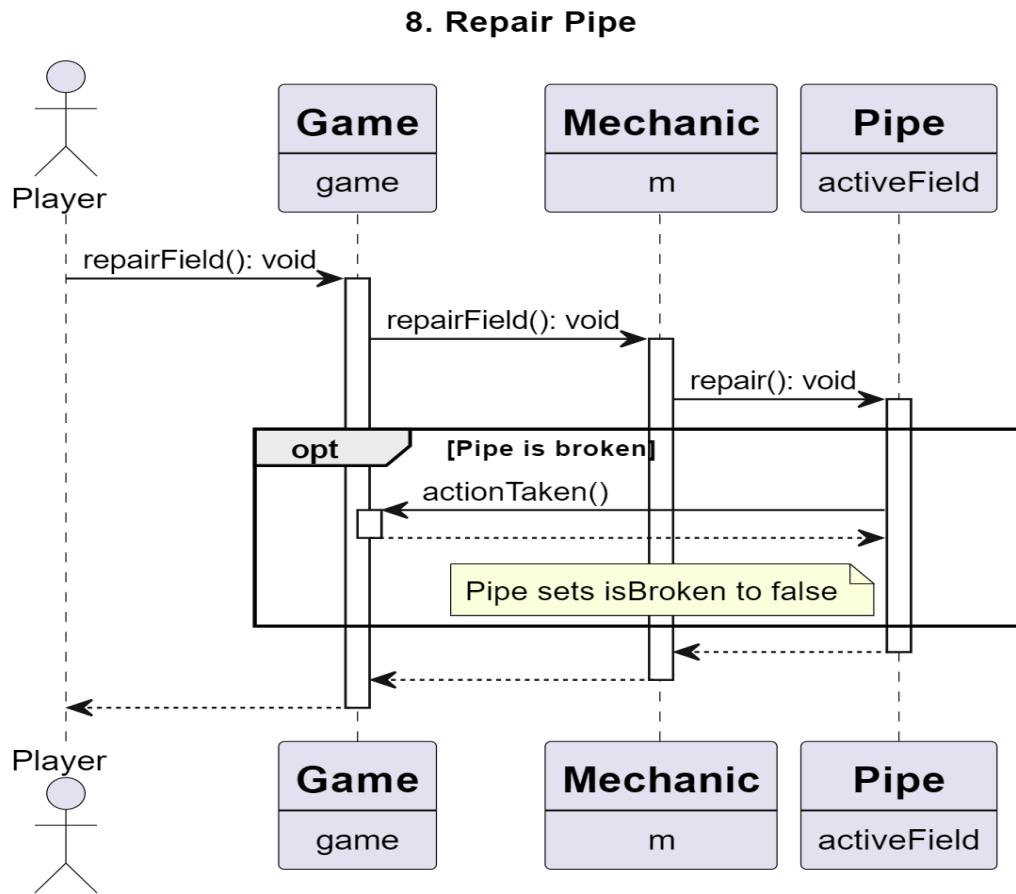
7. Break Pipe  
 a. Saboteur

### 7A. Break Pipe (Saboteur)



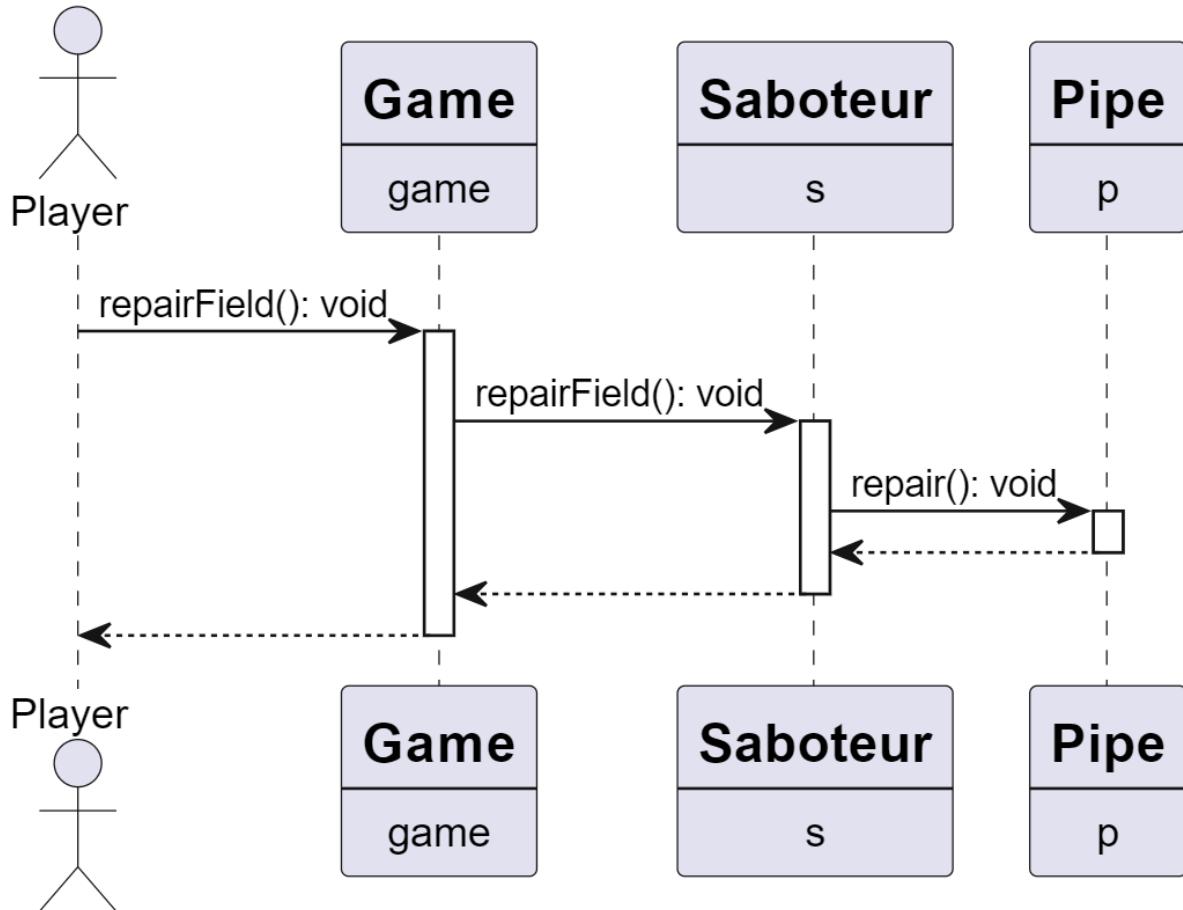
*b. Mechanic*

8. Repair Pipe  
 a. Mechanic



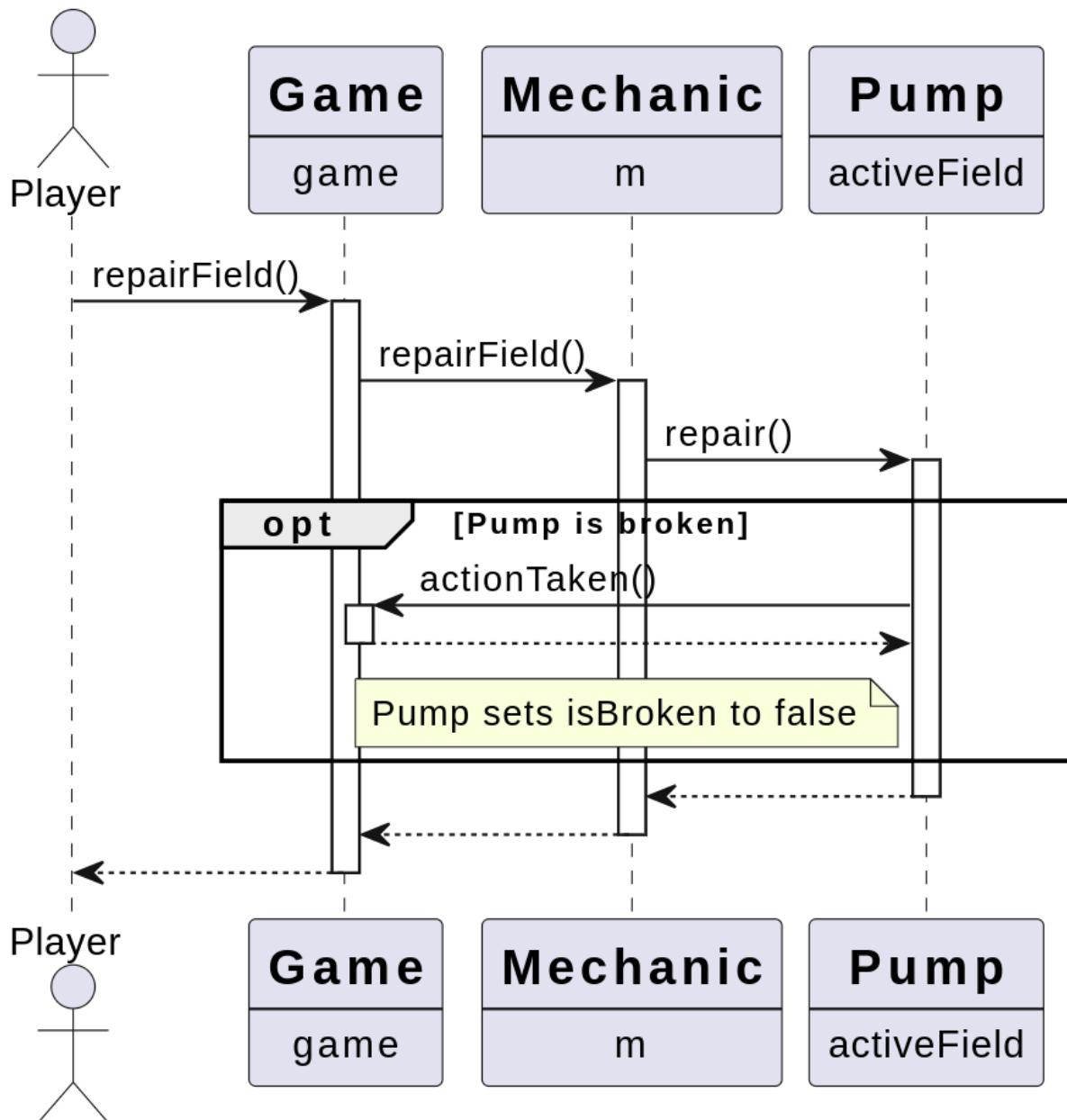
b. *Saboteur*

## 8.B Repair Pipe



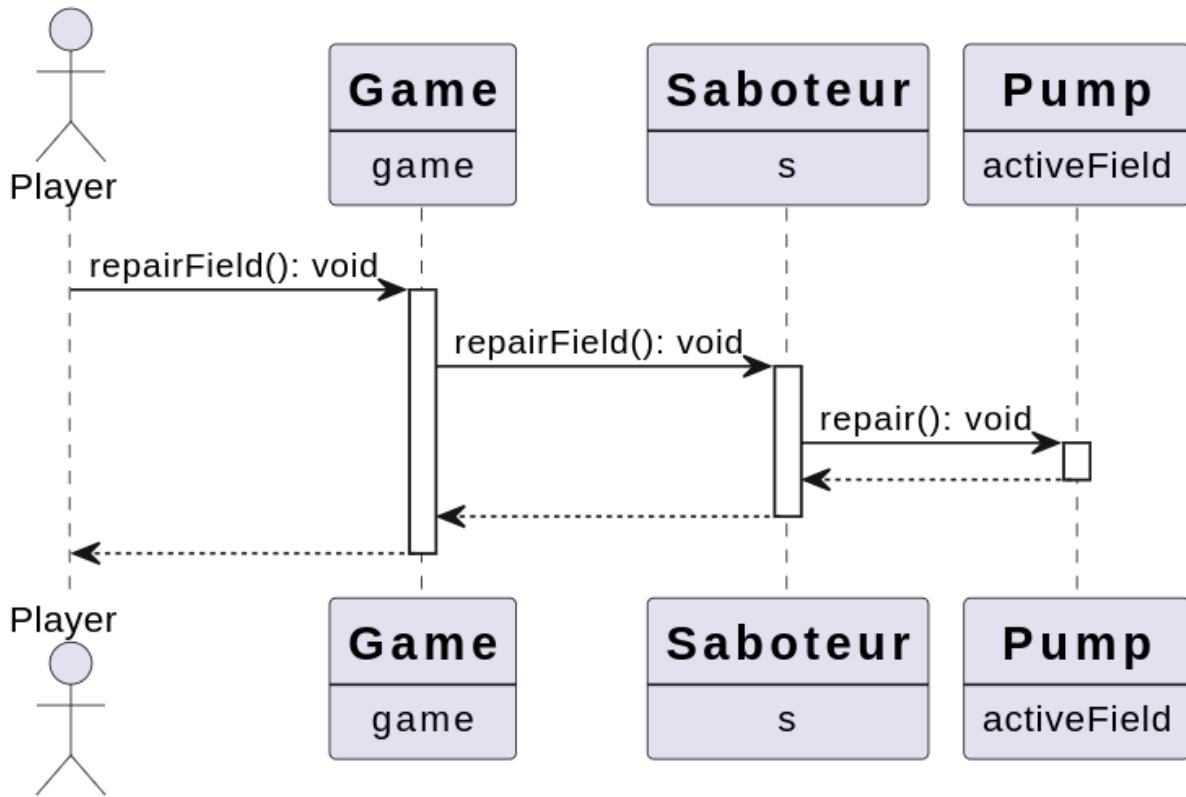
9. Repair Pump  
 a. Mechanic

## 9. Repair Pump



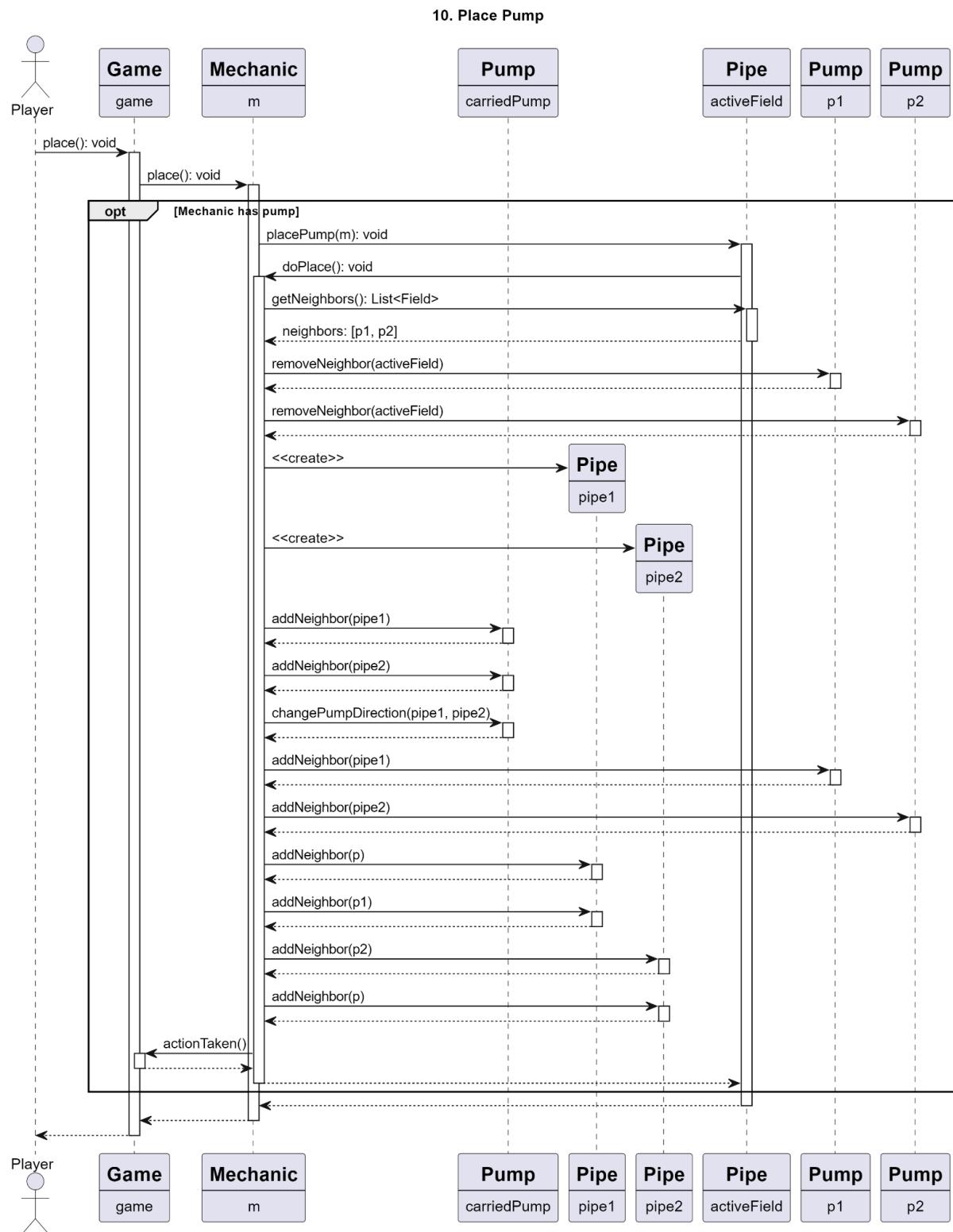
b. *Saboteur*

### 9.B Repair Pump



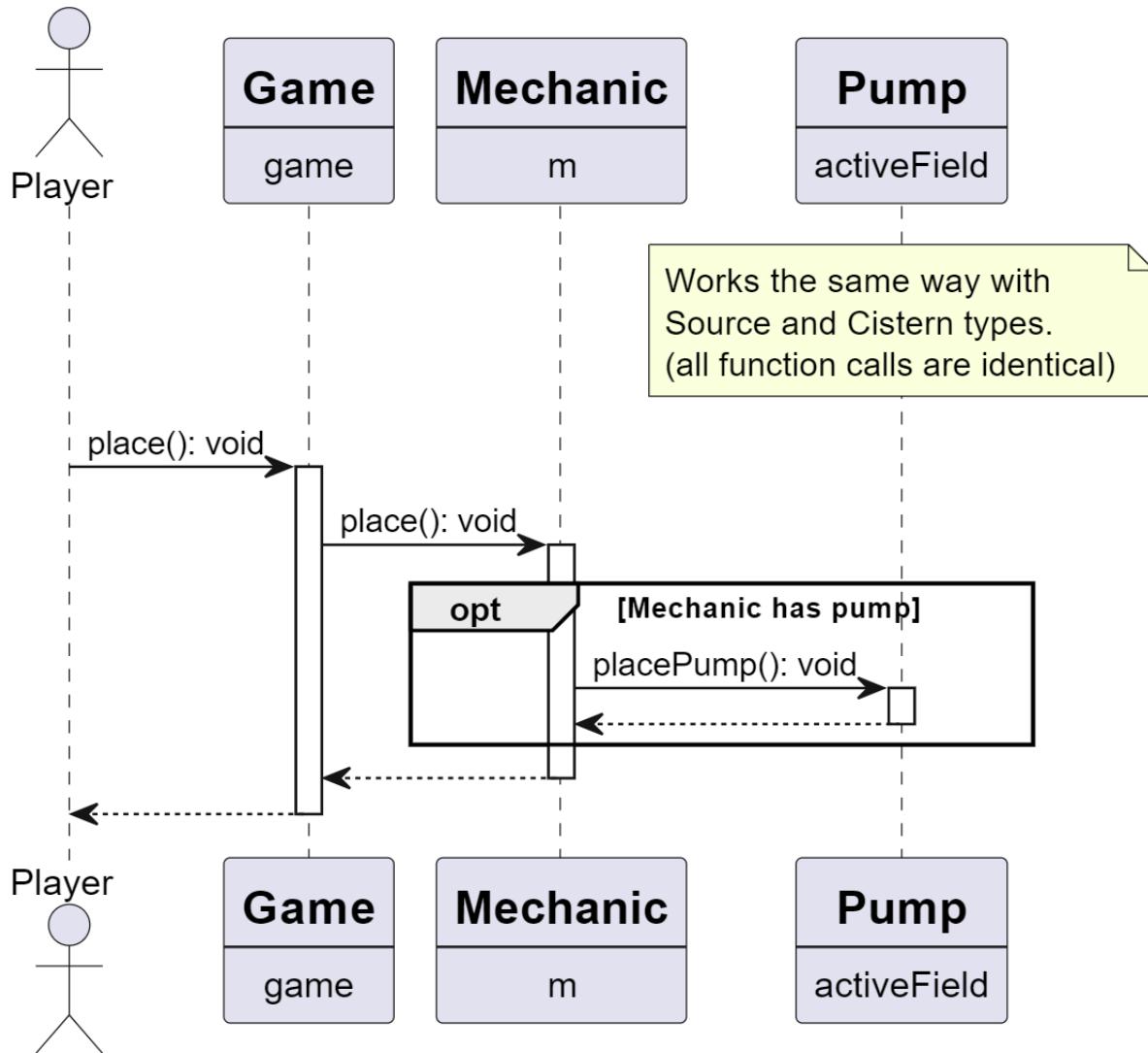
10. Place Pump

a. Mechanic



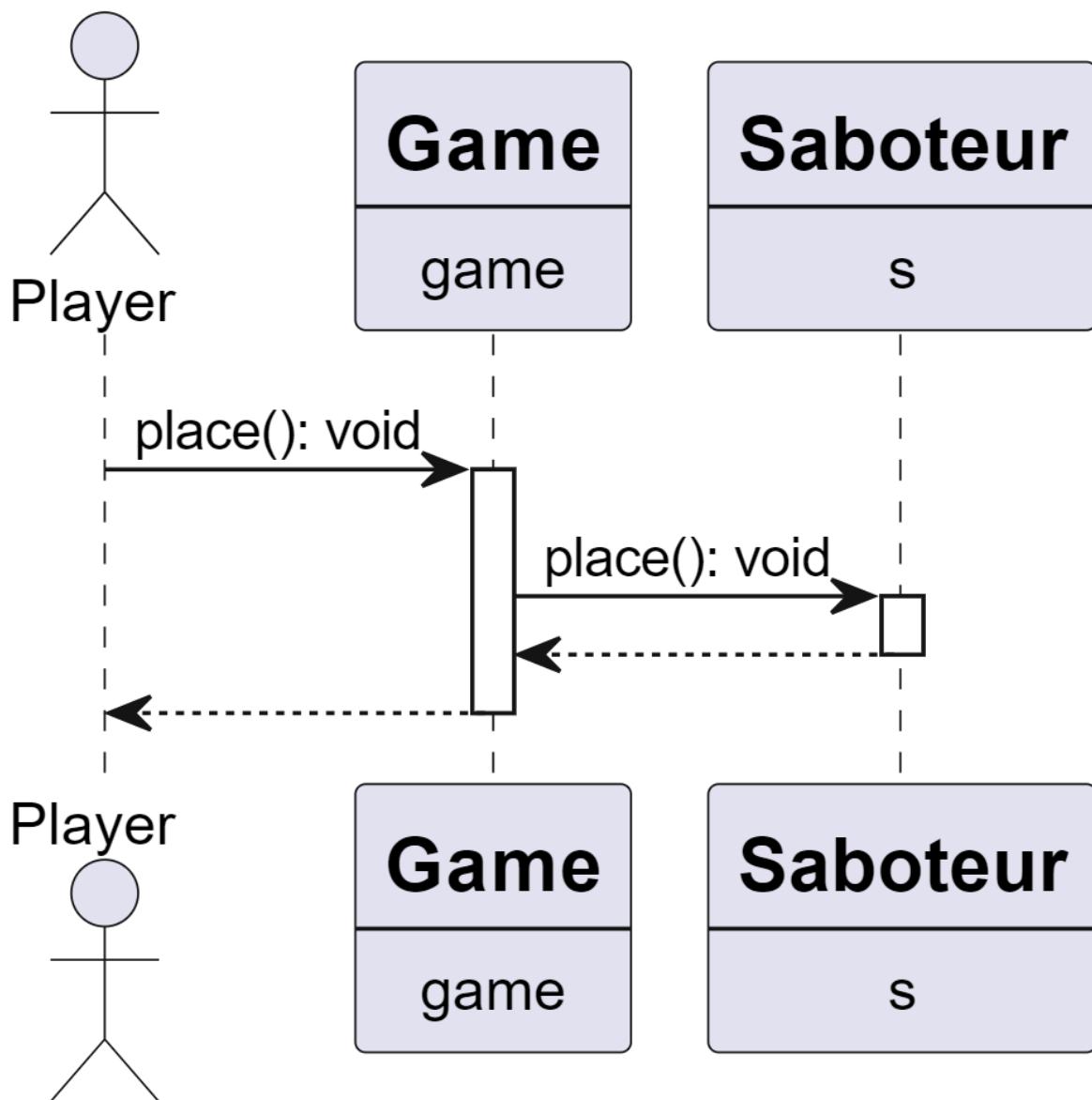
b. Mechanic on wrong Field

## 10.B Place Pump

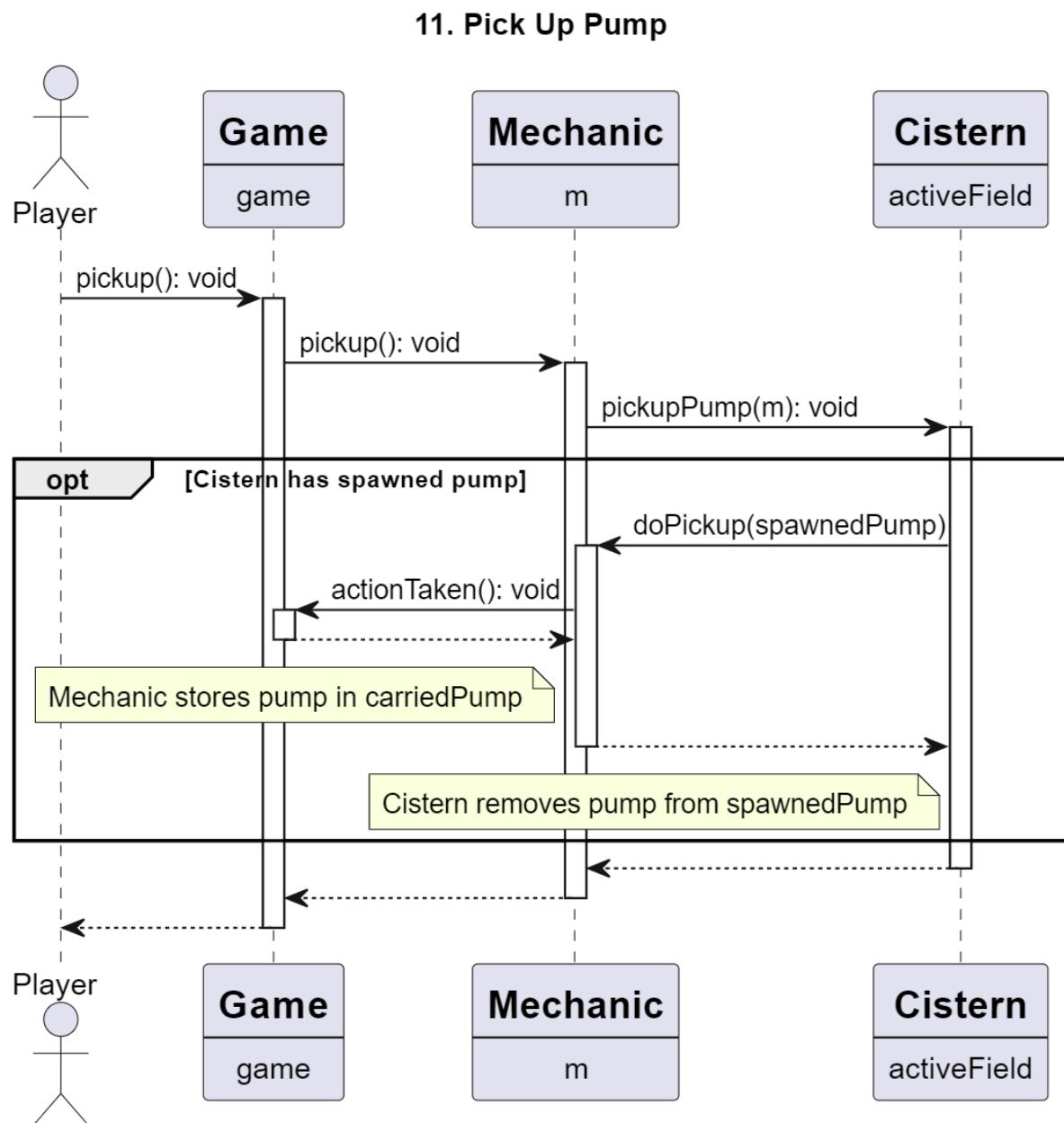


c. Saboteur

## 10.C Place Pump

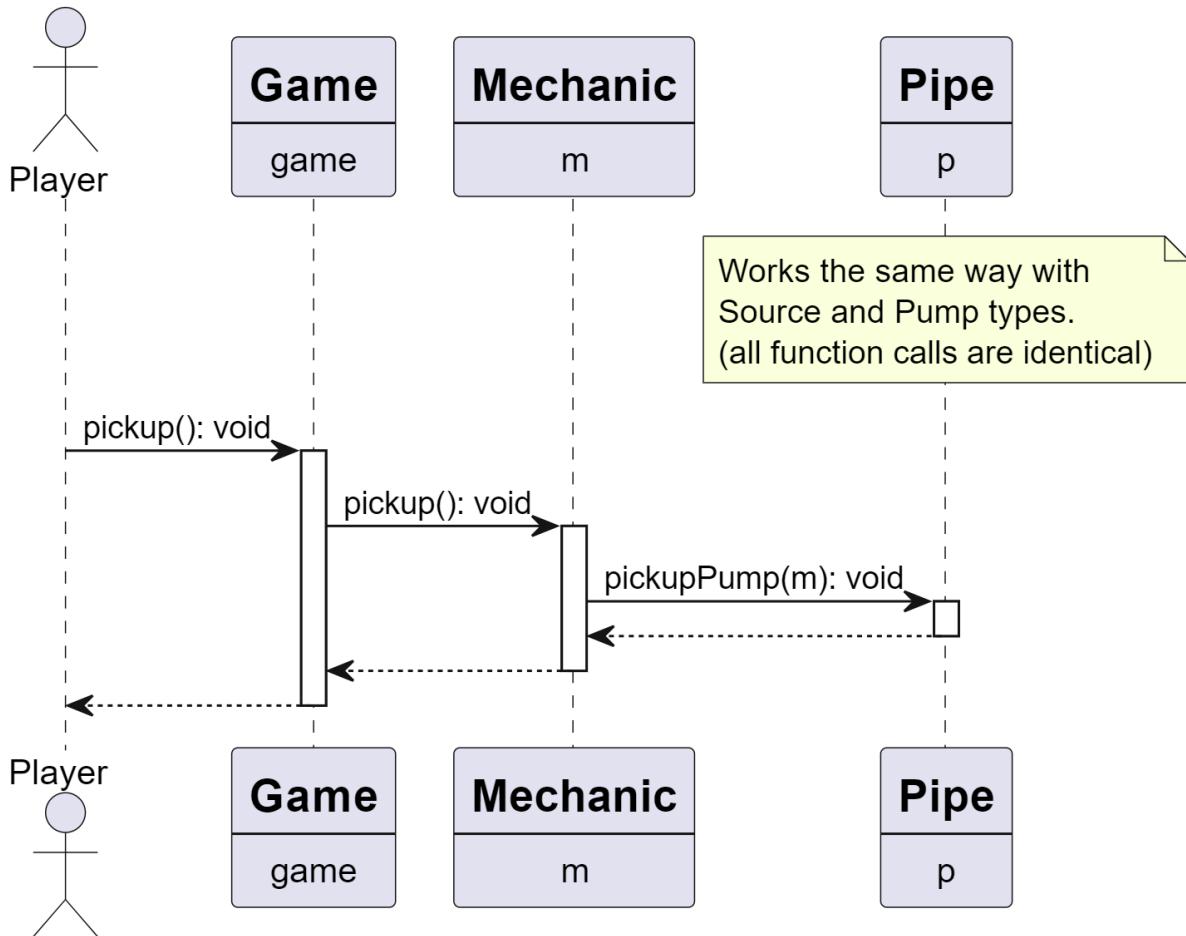


*11. Pick Up Pump*  
*a. Mechanic*



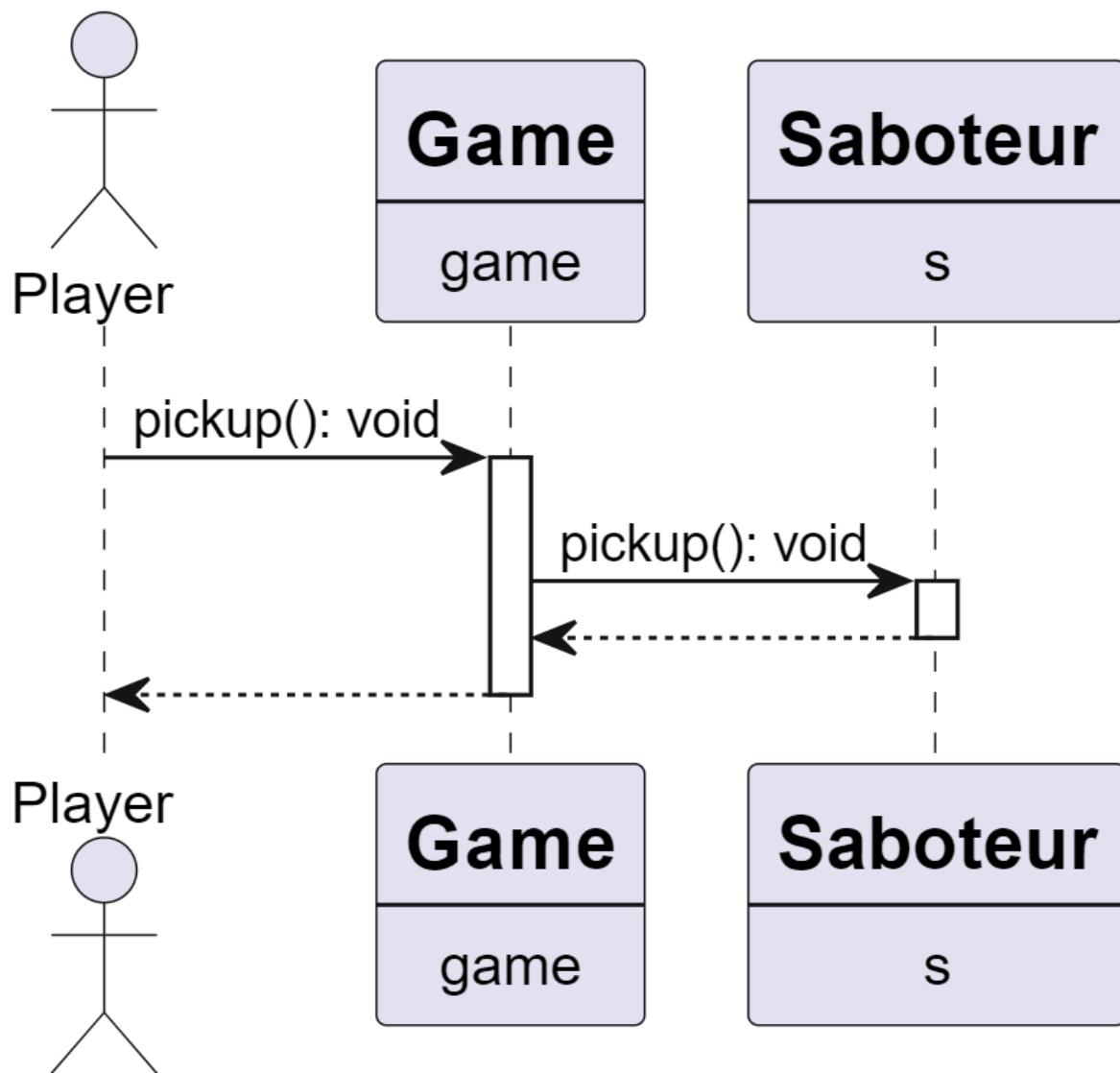
## b. Mechanic on wrong Field

## 11.B Pick Up Pump

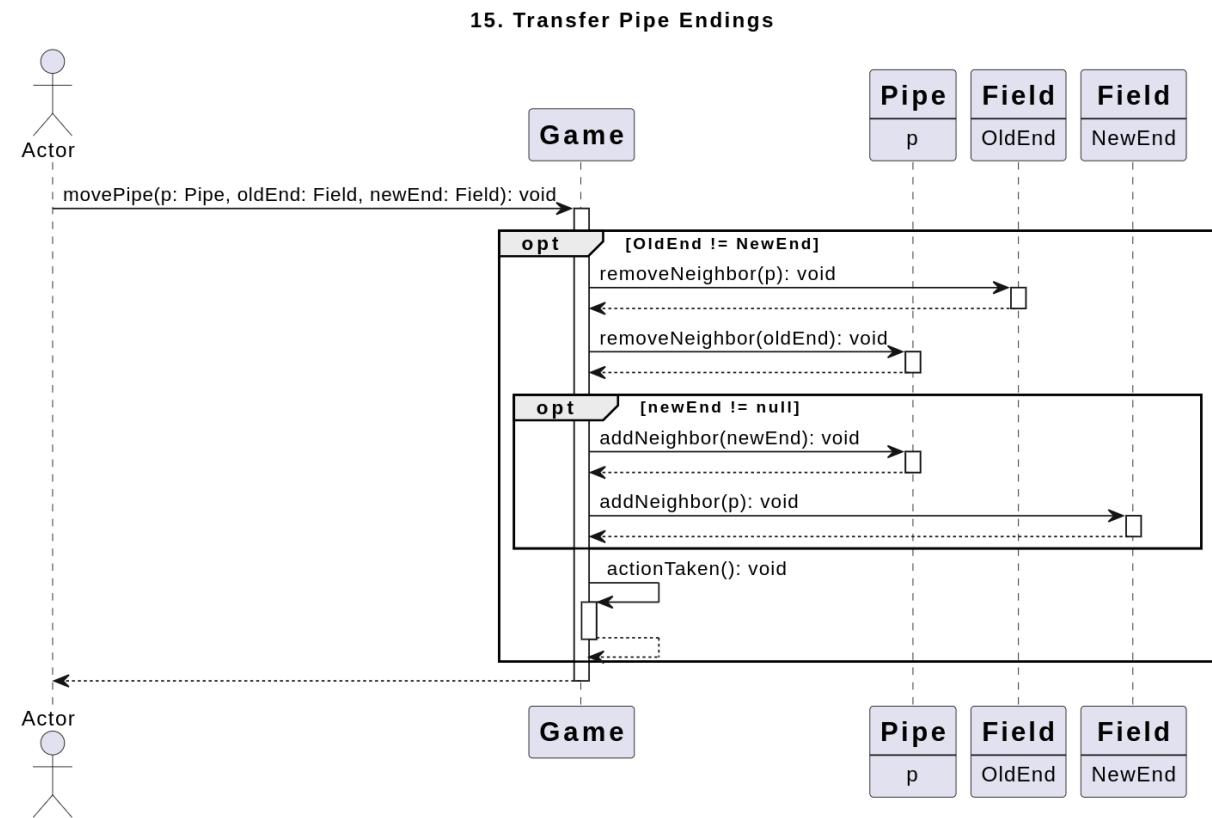


c. Saboteur

## 11.C Pick Up Pump

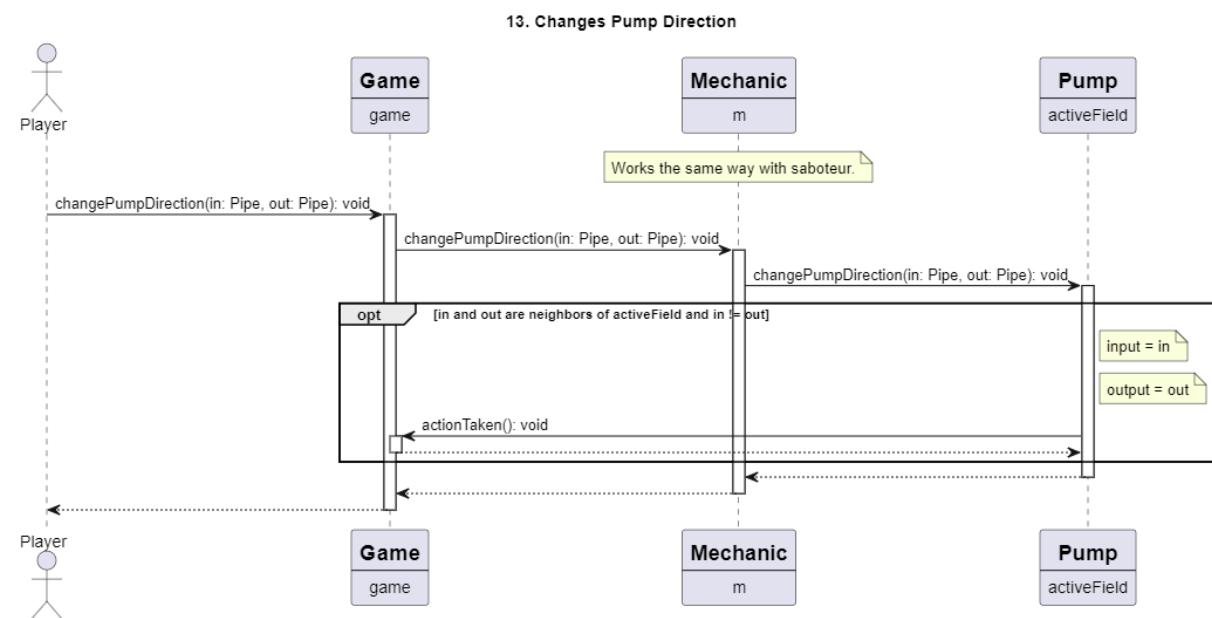


## 12. Transfer Pipe Endings

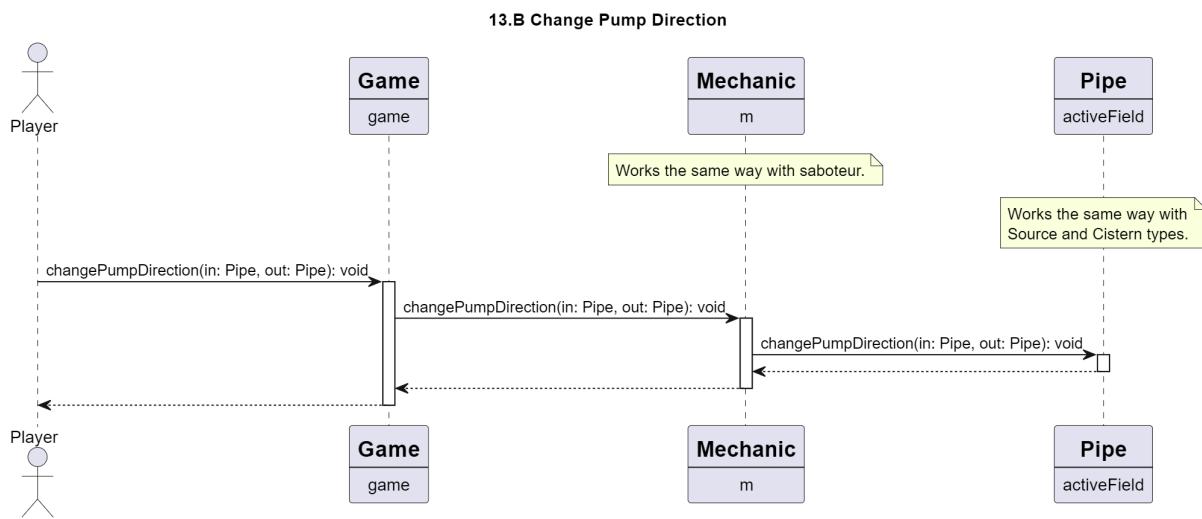


## 13. Change Pump Direction

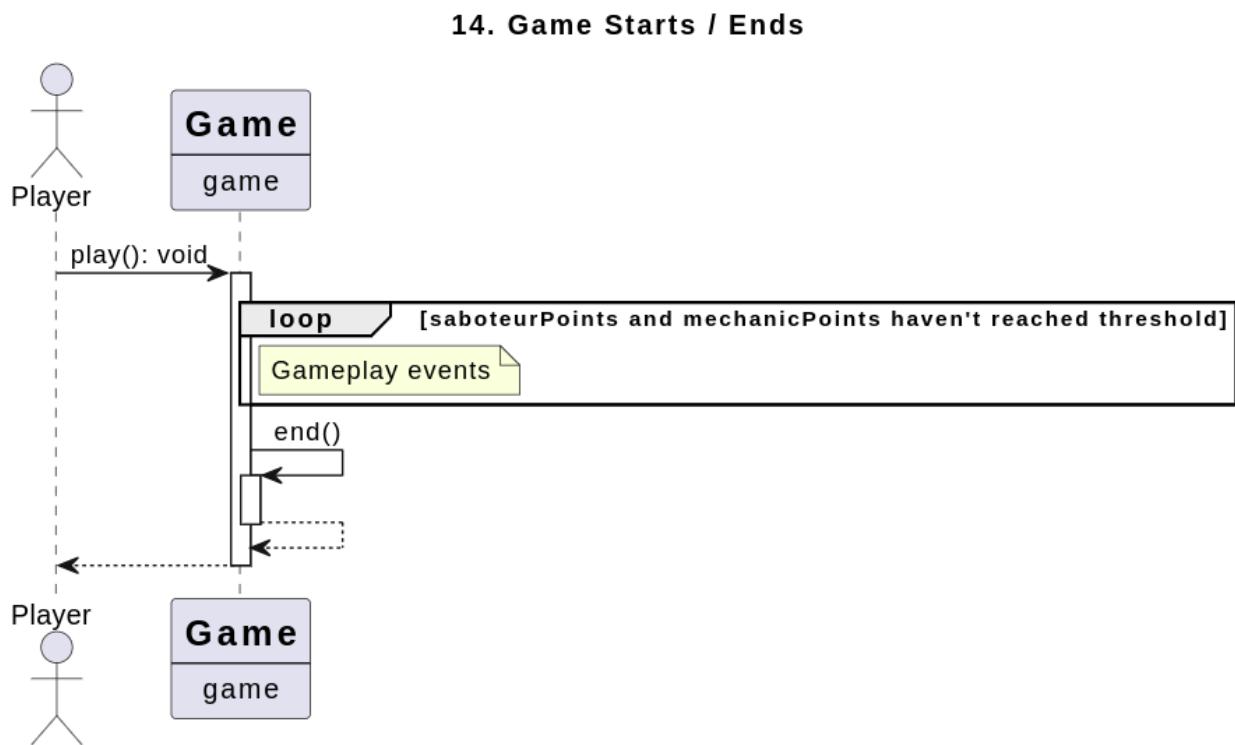
a. Correct



## b. Not on a Pump



## 14. Game Starts / Ends

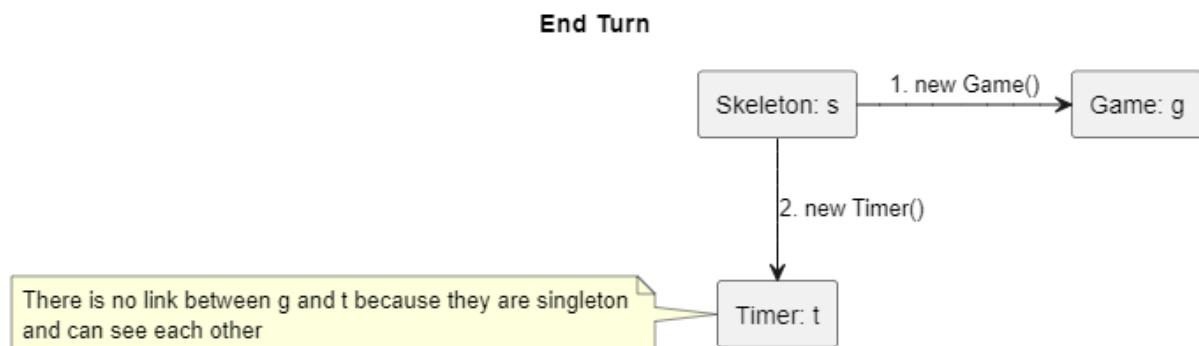


## 5.4 Kommunikációs diagramok

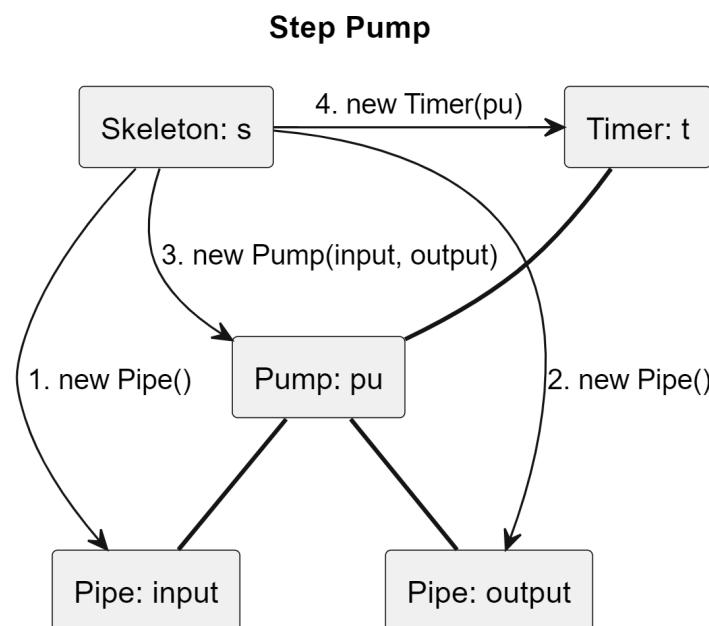
A diagramokon a szkeleton objektum és a többi objektum közé nem raktam kapcsolati linket, mert az nem tartozik a use-case objektumainak kapcsolataihoz, illetve így átláthatóbbak is a diagramok.

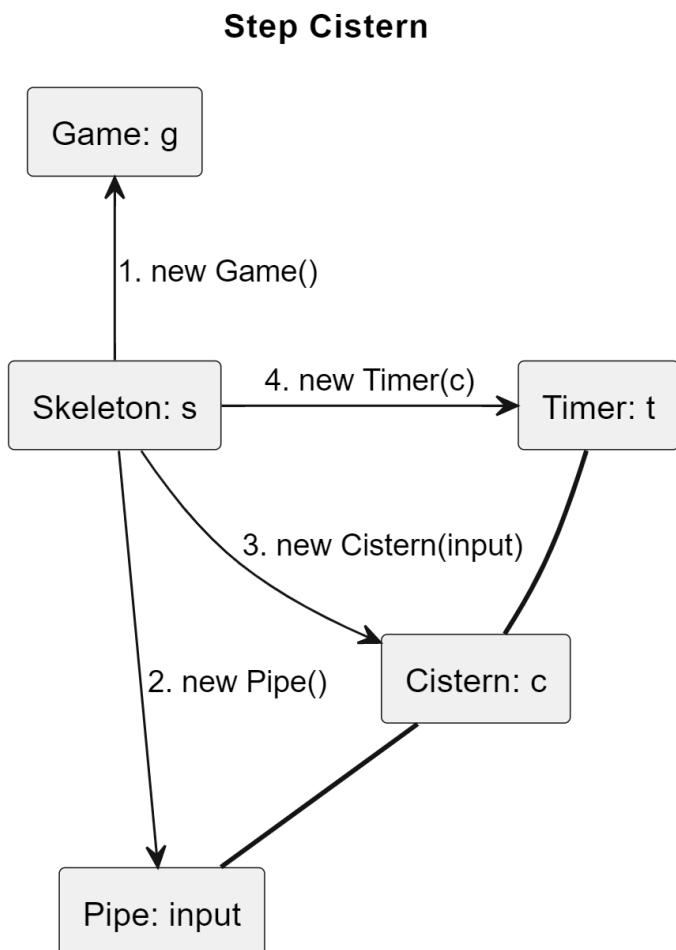
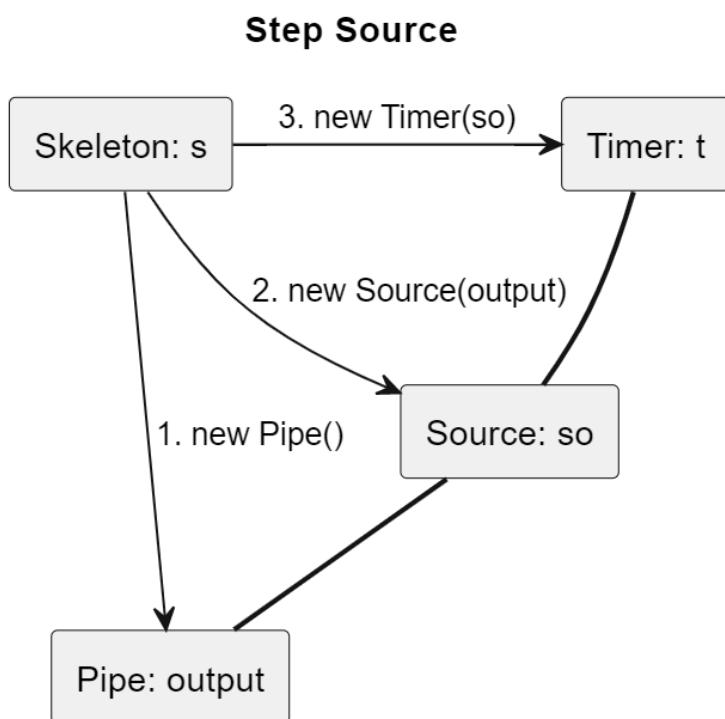
Vannak diagramok, melyeknél a számozásban betű is szerepel. A betűk az nem sorrendet jelölnek, mint a számok, hanem alternatív lefutásokat. Azt, hogy melyik nyílnak kell meghívódnia, az örfeltételben jelzem.

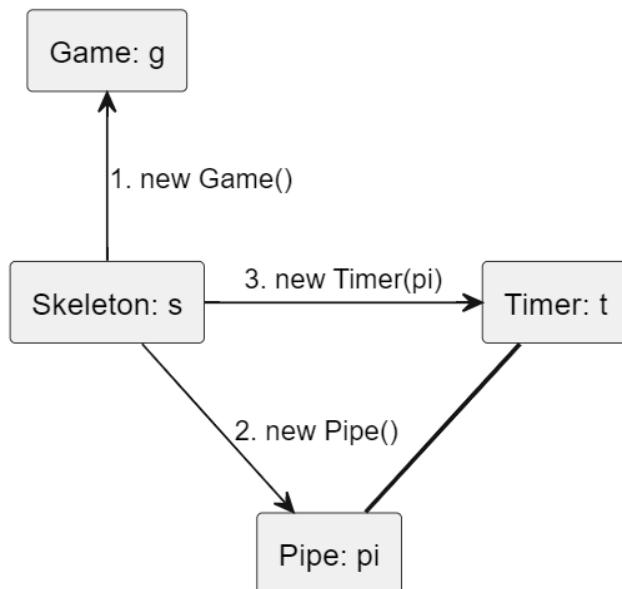
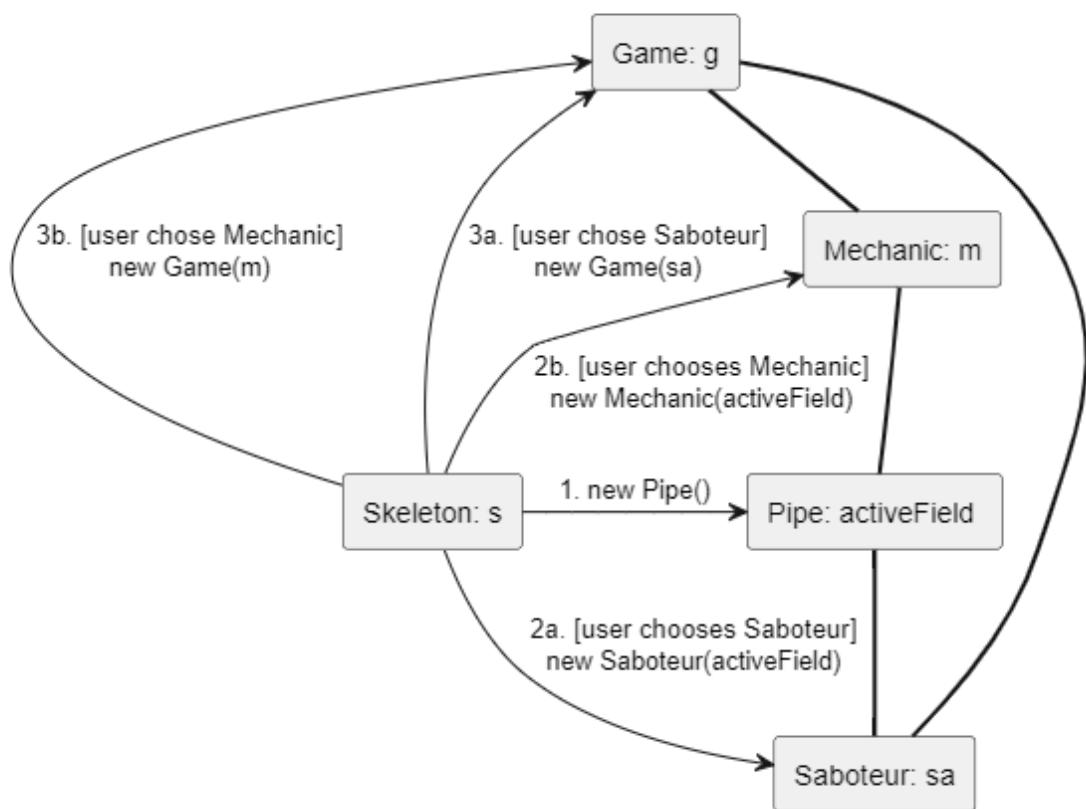
### Use-Case: End Turn



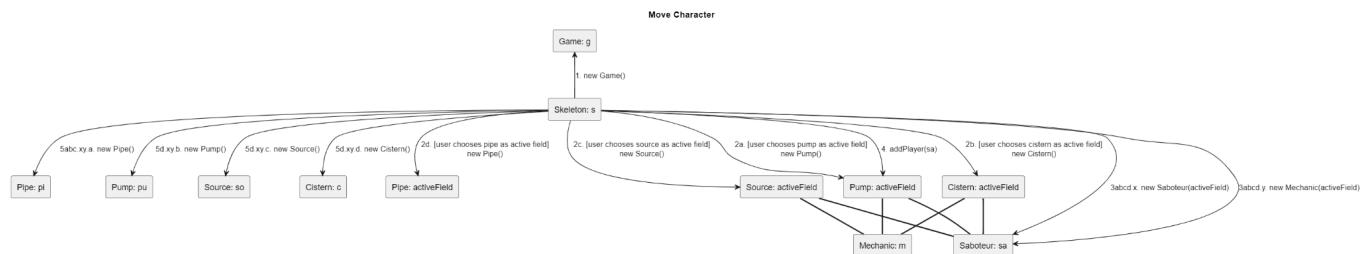
### Use-Case: Step Pump



**Use-Case: Step Cistern****Use-Case: Step Source**

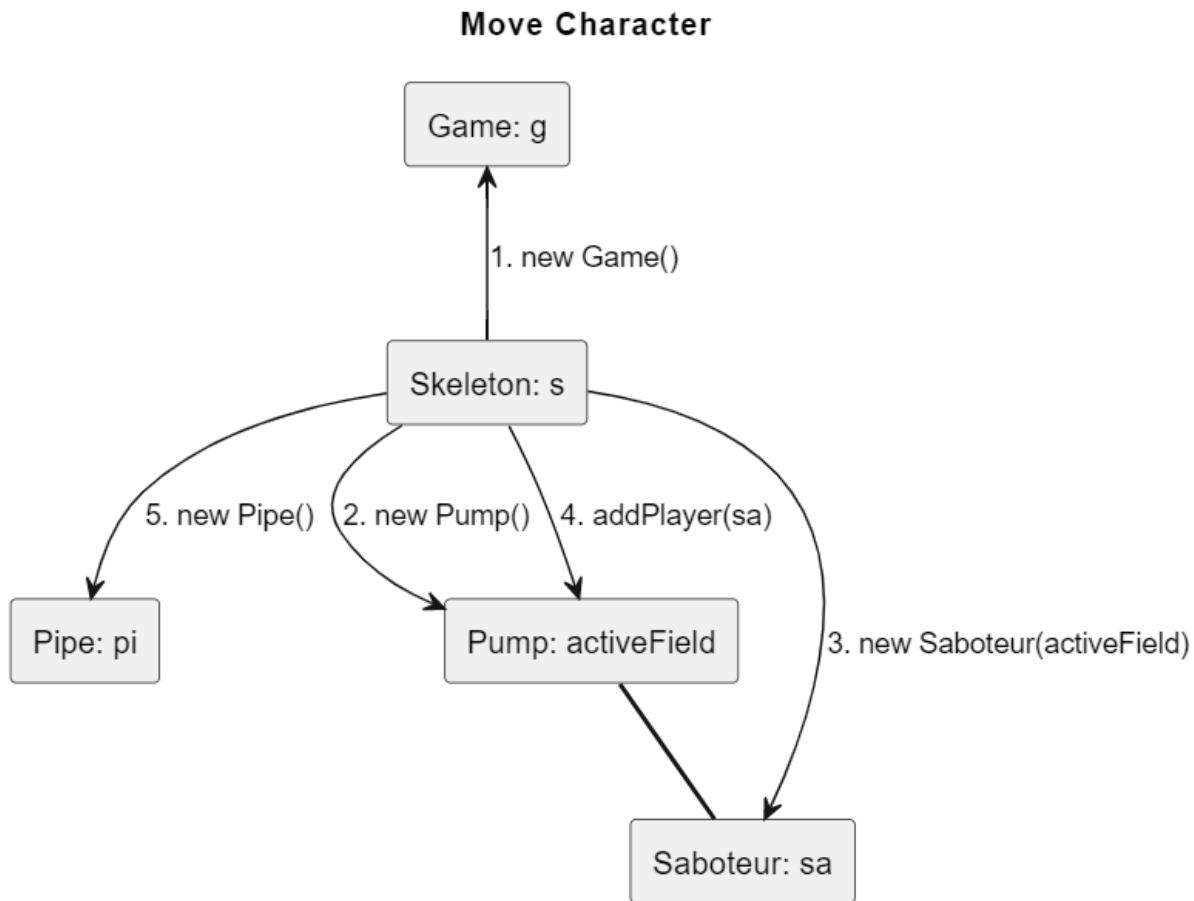
**Use-Case: Change Pipe State****Change Pipe State****Use-Case: Break Pipe****Use-Case: Repair Pipe****Break Pipe & Repair Pipe**

v

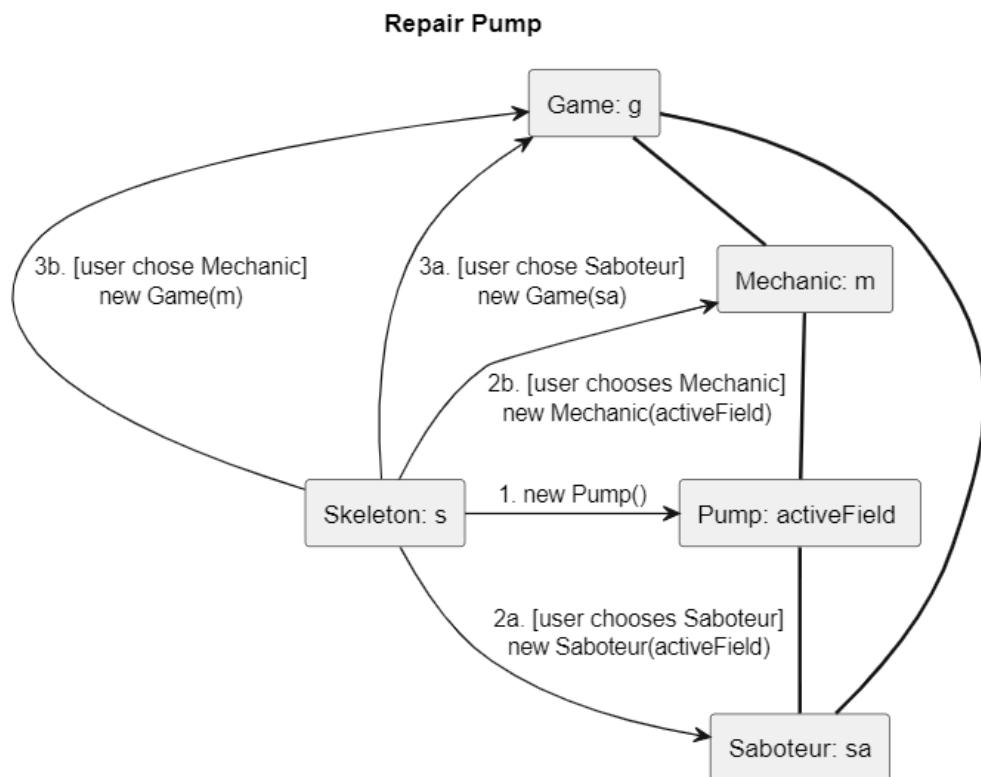
**Use-Case: Move Character**

Ennek a use-case-nek a sok alternatív forgatókönyve miatt a fenti átláthatatlan diagram jön létre, ezért ezt a működést inkább csak egy konkrét eseten ábrázolom.

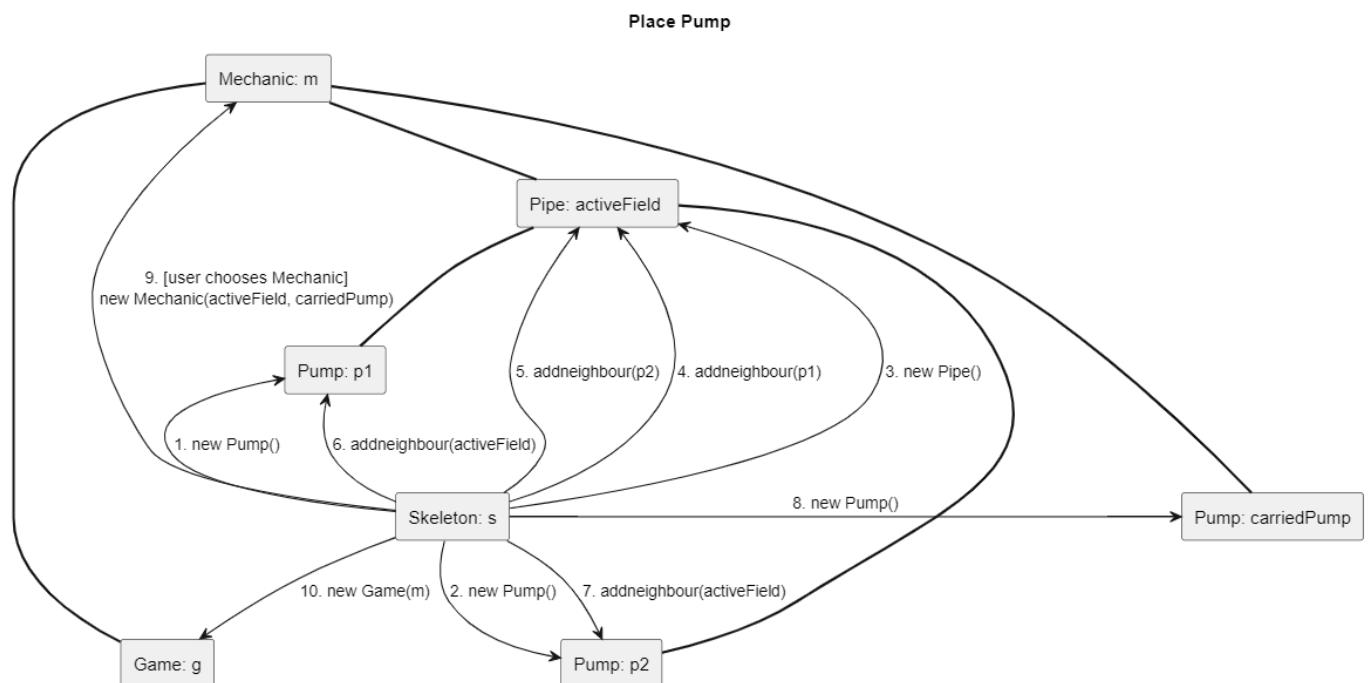
Az eset: Szabotör pumpáról lép csőre.

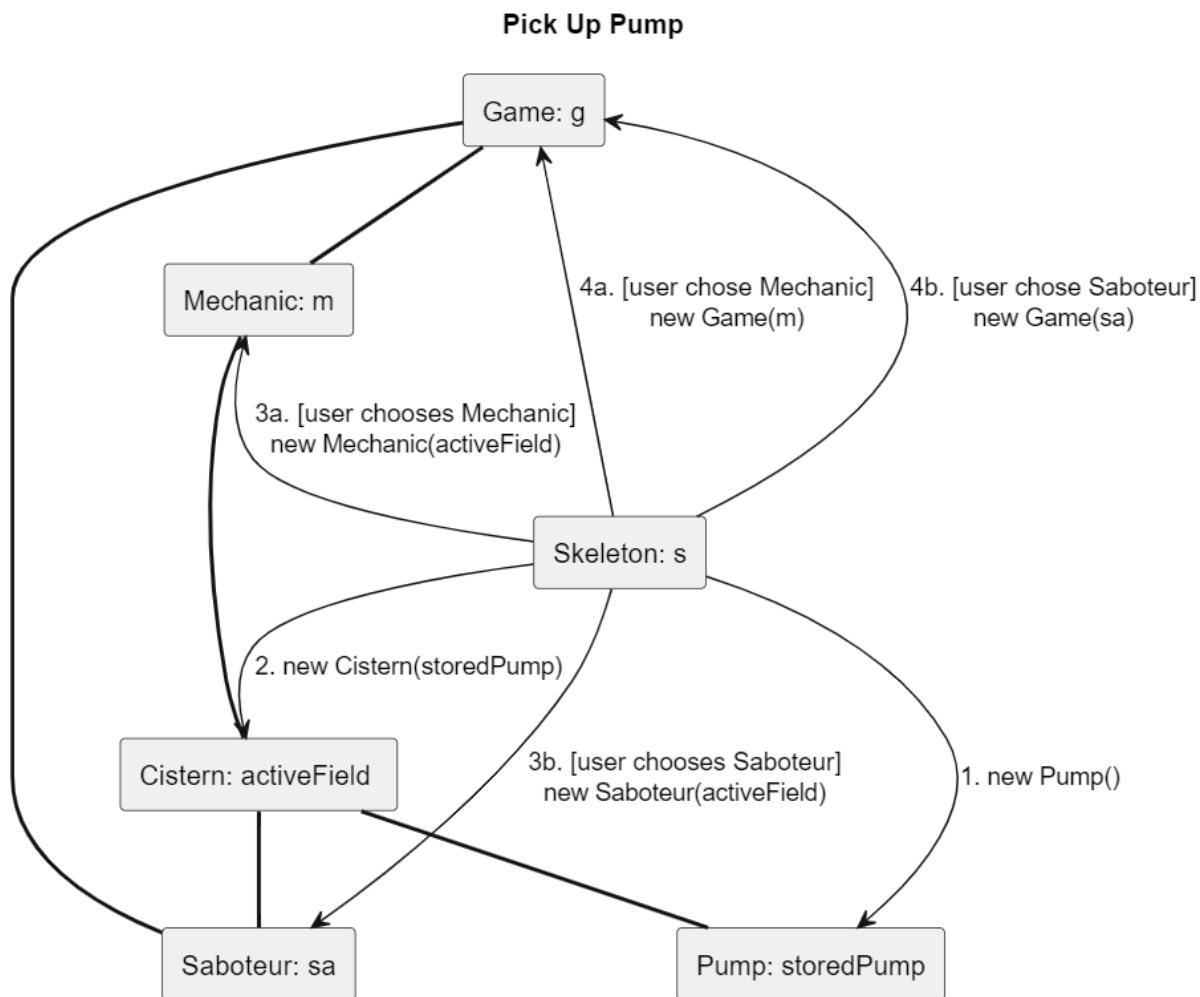
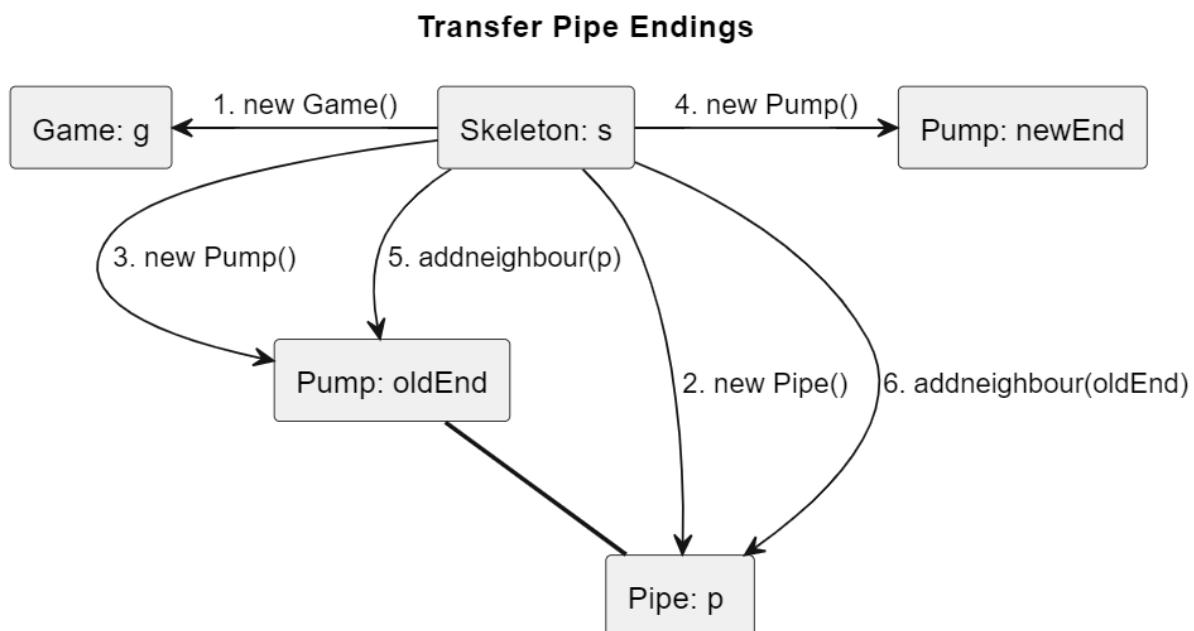


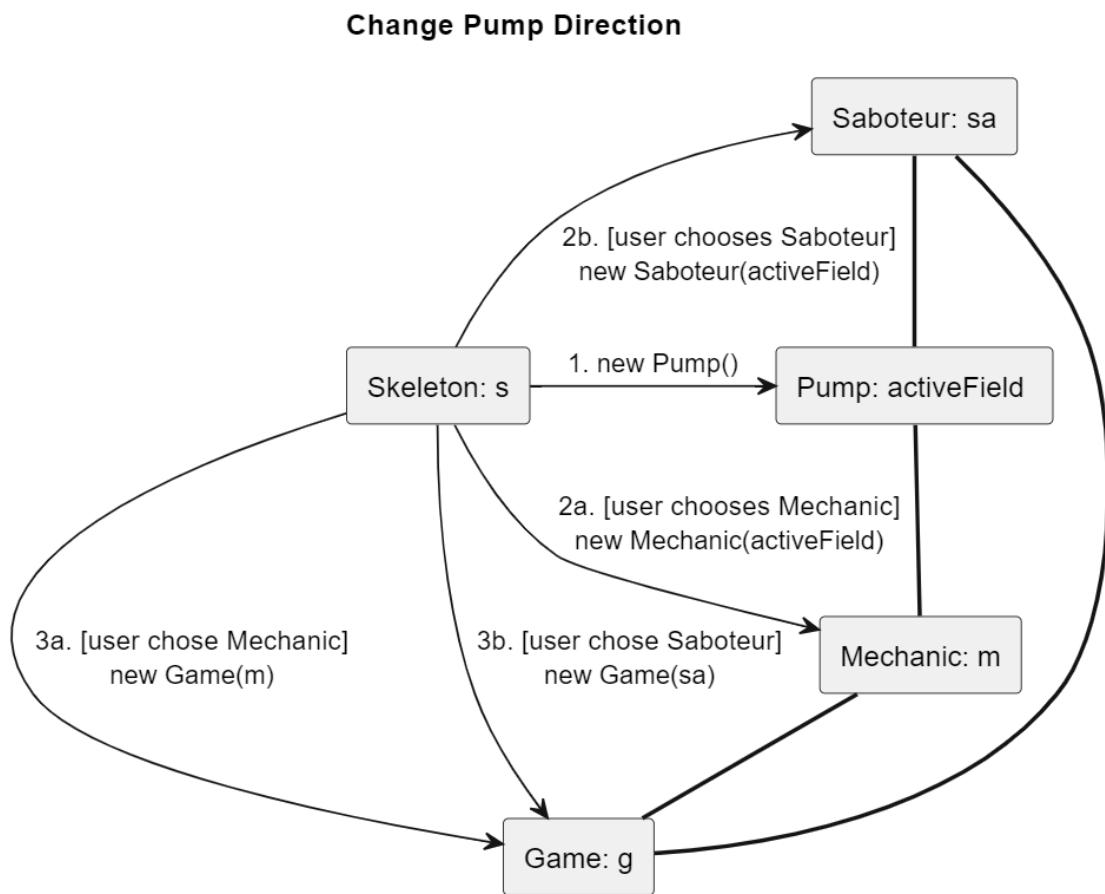
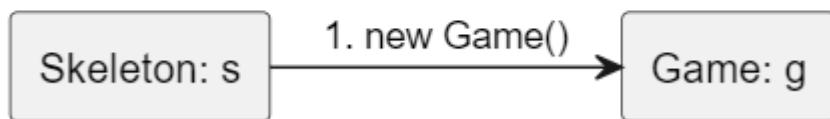
### Use-Case: Repair Pump



### Use-Case: Place Pump



**Use-Case: Pick Up Pump****Use-Case: Transfer Pipe Endings**

**Use-Case: Change Pump Direction****Use-Case: Game Starts / Ends****Game Starts / Ends**

## 5.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2023.03.31. 18:00	1.5 óra	Garai Mali Zavadil Varga	Értekezlet. Kérdések megfogalmazása, terv megbeszélése. Döntés: Zavadil javítja az analízis modellt.
2023.04.01. 11:00	1 óra	Zavadil	Analízis modell javítások
2023.04.01. 12:30	1 óra	Garai Mali Zavadil Varga Völgyesi	Értekezlet. Döntés: Mali és Varga csinálják a use-case-eket, forgatókönyveket. Garai csinálja a kommunikációs diagramokat. Völgyesi és Zavadil csinálják a szekvenciadiagramokat, Zavadil megírja az 5.2 fejezetet.
2023.04.01. 15:00	2 óra	Varga Mali	Use-case-ek megtervezése, use-case diagram elkészítése
2023.04.01. 17:00	3 óra	Varga	Use-case leírások megírása
2023.04.01. 21:00	1,5 óra	Varga	Use-case leírások kijavítása, és befejezése
2023.04.02. 10:15	5 óra	Völgyesi	Szekvencia diagramok készítése.
2023.04.02. 10:30	4.75 óra	Zavadil	Szekvenciadiagramok készítése
2023.04.02. 17:00	1 óra	Zavadil	5.2 fejezet megírása
2023.04.02. 19:00	2 óra	Garai	Kommunikációs diagramok tervezése
2023.04.03. 7:00	4 óra	Garai	Kommunikációs diagramok elkészítése

## 6. Szkeleton beadás

### 6.1 Fordítási és futtatási útmutató

#### 6.1.1 Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
Cistern.java	1,901 byte	2023.04.16. 10:25:37 PM CEST	A Cistern osztály megvalósítása
Field.java	4,339 byte	2023.04.16. 10:25:37 PM CEST	A Field osztály megvalósítása
Game.java	5,759 byte	2023.04.16. 10:25:37 PM CEST	A Game osztály megvalósítása
Main.java	97 byte	2023.04.16. 10:25:37 PM CEST	A program belépési pontja, elindítja Skeleton osztály inicializálását
Mechanic.java	5,096 byte	2023.04.16. 10:25:37 PM CEST	A Mechanic osztály megvalósítása
Periodic.java	317 byte	2023.04.16. 10:25:37 PM CEST	A Periodic interfész megvalósítása
Pipe.java	5,591 byte	2023.04.16. 10:25:37 PM CEST	A Pipe interfész megvalósítása
Player.java	2,276 byte	2023.04.16. 10:25:37 PM CEST	A Player osztály megvalósítása
Pump.java	3,265 byte	2023.04.16. 10:25:37 PM CEST	A Pump osztály megvalósítása
Saboteur.java	1,303 byte	2023.04.16. 10:25:37 PM CEST	A Saboteur osztály megvalósítása
Skeleton.java	24,606 byte	2023.04.16. 10:25:37 PM CEST	A Skeleton osztály kódja. Ennek a feladata a use-case kiválasztás lehetővé tétele és ezekhez az objektumok létrehozása és a szekvencia diagramok szerinti viselkedés elindítása
Source.java	521 byte	2023.04.16. 10:25:37 PM CEST	A Source osztály megvalósítása
Stateful.java	192 byte	2023.04.16. 10:25:37 PM CEST	A Stateful interfész megvalósítása
Timer.java	2,054 byte	2023.04.16. 10:25:37 PM CEST	A Timer osztály megvalósítása

### 6.1.2 Fordítás

A 6.1.1 listában leírt összes fájlt (és csak ezeket) helyezzük bele egy üres mappába. Ezután ellenőrizzük, hogy a java és javac parancsok fel vannak-e telepítve.

Ezután adjuk ki a következő parancsot a fordításhoz:

```
javac *.java
```

### 6.1.3 Futtatás

A 6.1.2. dokumentumban leírt lépések után ugyanabban a mappában adjuk ki a következő parancsot a futtatáshoz:

```
java Main
```

Ezt követően a program elindul.

## 6.2 Értékelés

Tag neve	Tag neptun	Munka százalékban
Mali Bence	ZA2ENI	20
Zavadil Balázs Dávid	FOIFDF	20
Völgyesi Soma	GBI4MD	20
Varga David Imre	Z8JXEQ	20
Garai Donát Róbert	CRSL00	20

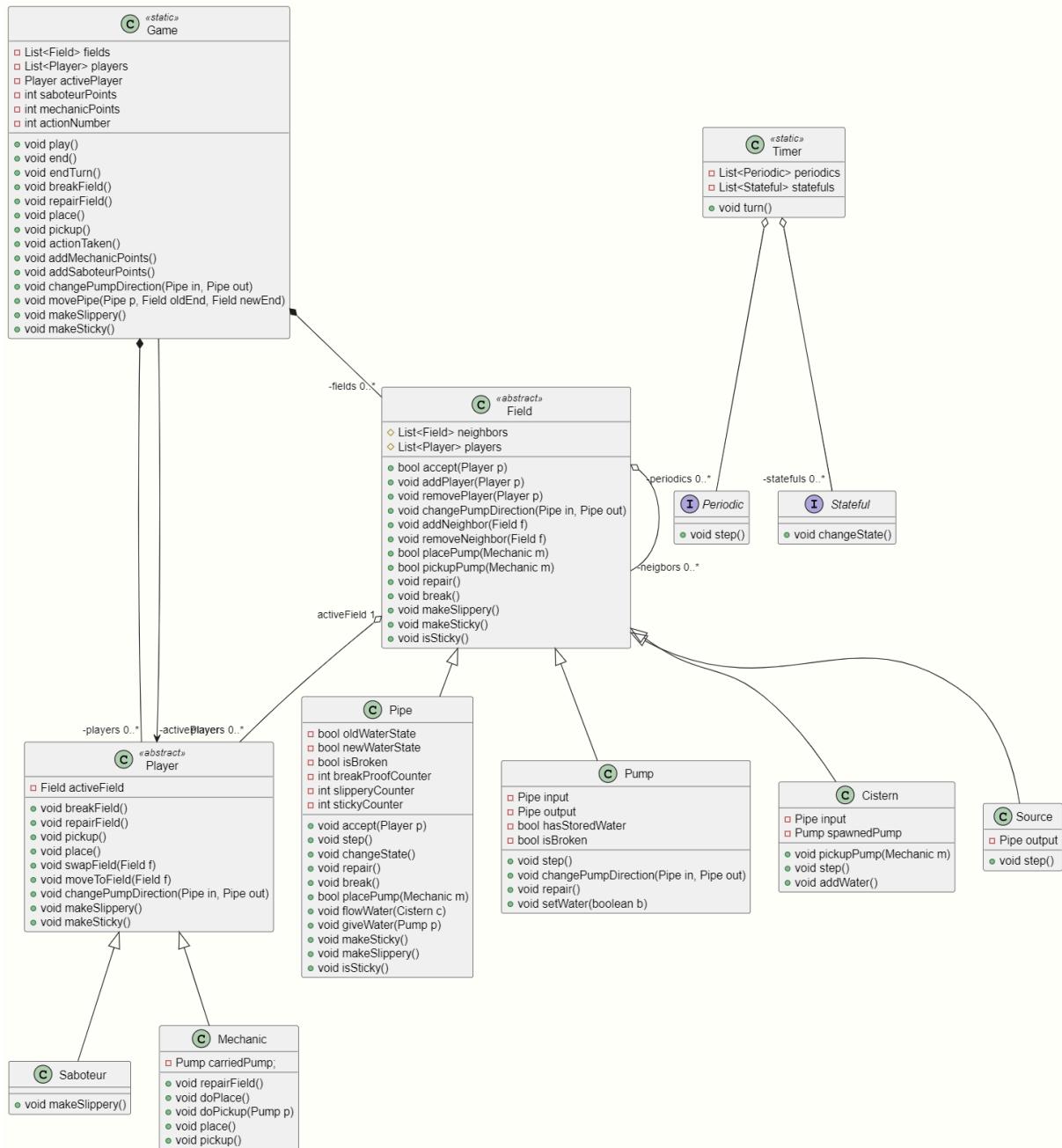
### 6.3 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2023.04.13. 14:00	3 óra	Garai Mali Völgyesi Varga Zavadil	Értekezlet: Feladatok átbeszélése és kiosztása
2023.04.13. 14:00	3 óra	Zavadil	Szkeleton kiíró függvényei és javadoc commentek az osztályokhoz
2023.04.14 17:00	2 óra	Varga	Use case-ek megírásának iniciálizálása. Az IntelliJ skeleton osztályba függvények deklarálása
2023.04.15. 10:00	5 óra	Garai	Szkeleton menüjének, use-case-ek kiválasztásához és inicializálásához szükséges függvények megírása, szükséges konstruktörök megírása.
2023.04.15. 10:00	2 óra	Völgyesi	Osztályok megírása
2023.04.15. 16:00	6 óra	Mali	Első 7 use case-hez szükséges metódusok megírása
2023.04.16 11:00	6 óra	Varga	Az előző use casek megírása
2023.04.16. 16:30	6.5 óra	Zavadil	Hibák javítása, tesztelés (lokális és vm-beli), dokumentum megírása
2023.04.16. 17:00	5 óra	Garai	Hibák javítása, tesztelés
2023.04.16 21:00	1,5 óra	Varga	Hibák javítása, tesztelés

## 7. Prototípus koncepciója

### 7.0 Változás hatása a modellre

#### 7.0.1 Módosult osztálydiagram



#### 7.0.2 Új vagy megváltozó metódusok

##### Player

- void breakField(): Meghívja az activeField tagváltozón a break() metódust.
- void makeSlippery(): A metódus implementációja ebben az osztályban üres.

- void makeSticky(): A karakter aktuális mezőjét ragadóssá teszi. Meghívja az activeField() tagváltozón a makeSticky() metódust.
- void moveToField(Field f): Továbbra is megpróbál átlépni a paraméterként kapott mezőre, de előtte az activeField tagváltozóján meghívja az isSticky() metódust és csak akkor lép, ha False-t kap vissza.

### Saboteur

- void makeSlippery(): A karakter aktuális mezőjét csúszóssá teszi. Meghívja az activeField() tagváltozón a makeSlippery() függvényt.

### Pipe

- void repair(): Továbbra is beállítja az isBroken értékét hamisra, és ha igaz volt meghívja a Game osztály actionTaken() metódusát. A változtatás hatására most a breakProofCounter értékét egy véletlenszerű pozitív értékre állítja.
- void break(): Mostantól csak akkor állítja az isBroken-t True-ra, ha a breakProofCounter értéke 0.
- void step(): A breakProofCounter, a slipperyCounter és a stickyCounter értékét csökkenti 1-gyel, ha még nem 0 az értékük.
- void makeSlippery(): A slipperyCounter értékét 5-re állítja és meghívja a Game osztály actionTaken() metódusát.
- void makeSticky(): A stickyCounter értékét 5-re állítja és meghívja a Game osztály actionTaken() metódusát.
- bool isSticky(): A metódus implementációja ebben az osztályban False-szal tér vissza, ha a stickyCounter 0-val egyenlő, és True-val ha nagyobb, mint 0.

### Field

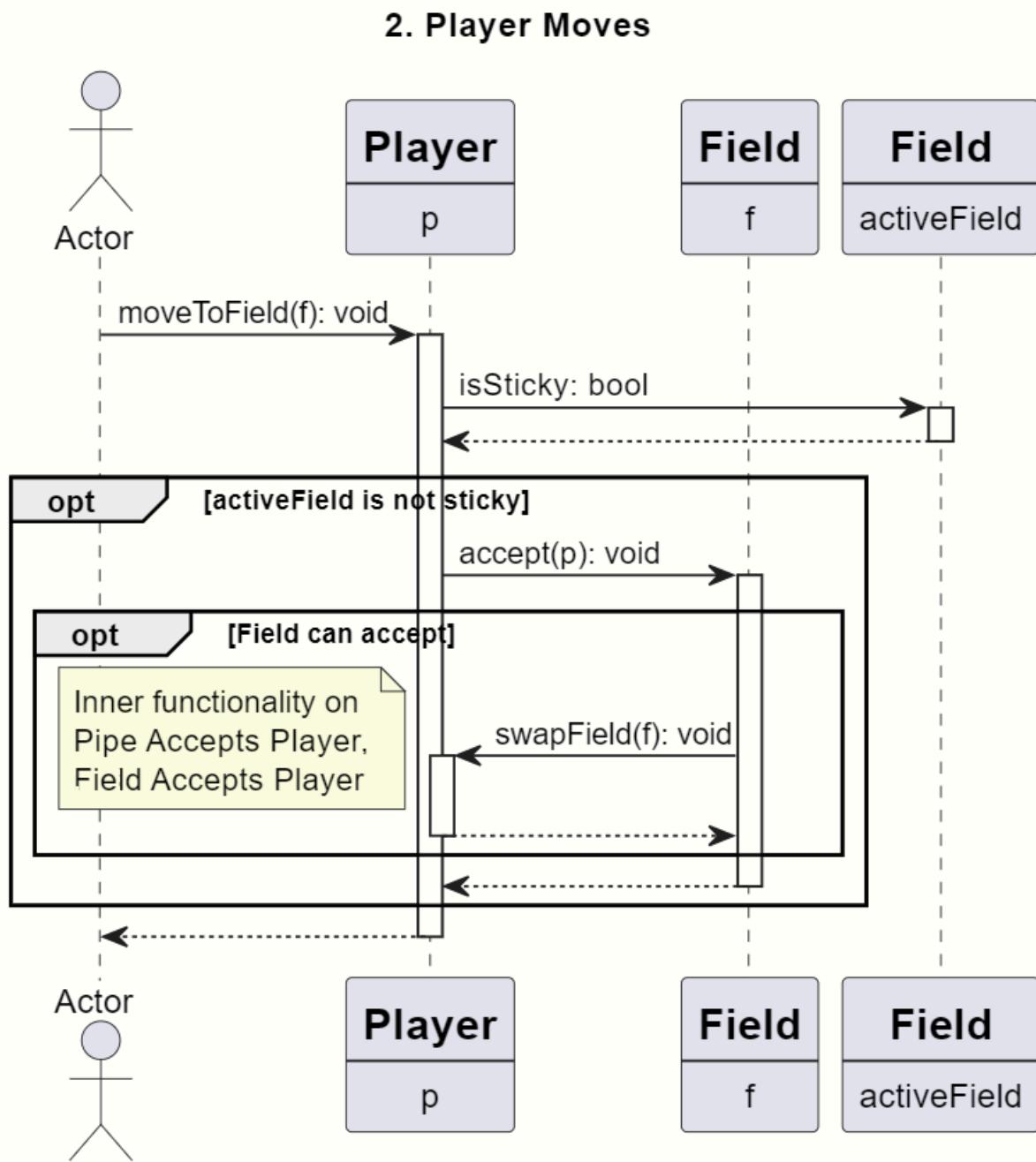
- void makeSlippery(): A metódus implementációja ebben az osztályban üres.
- void makeSticky(): A metódus implementációja ebben az osztályban üres.
- bool isSticky(): A metódus implementációja ebben az osztályban minden False-szal tér vissza.

### Game

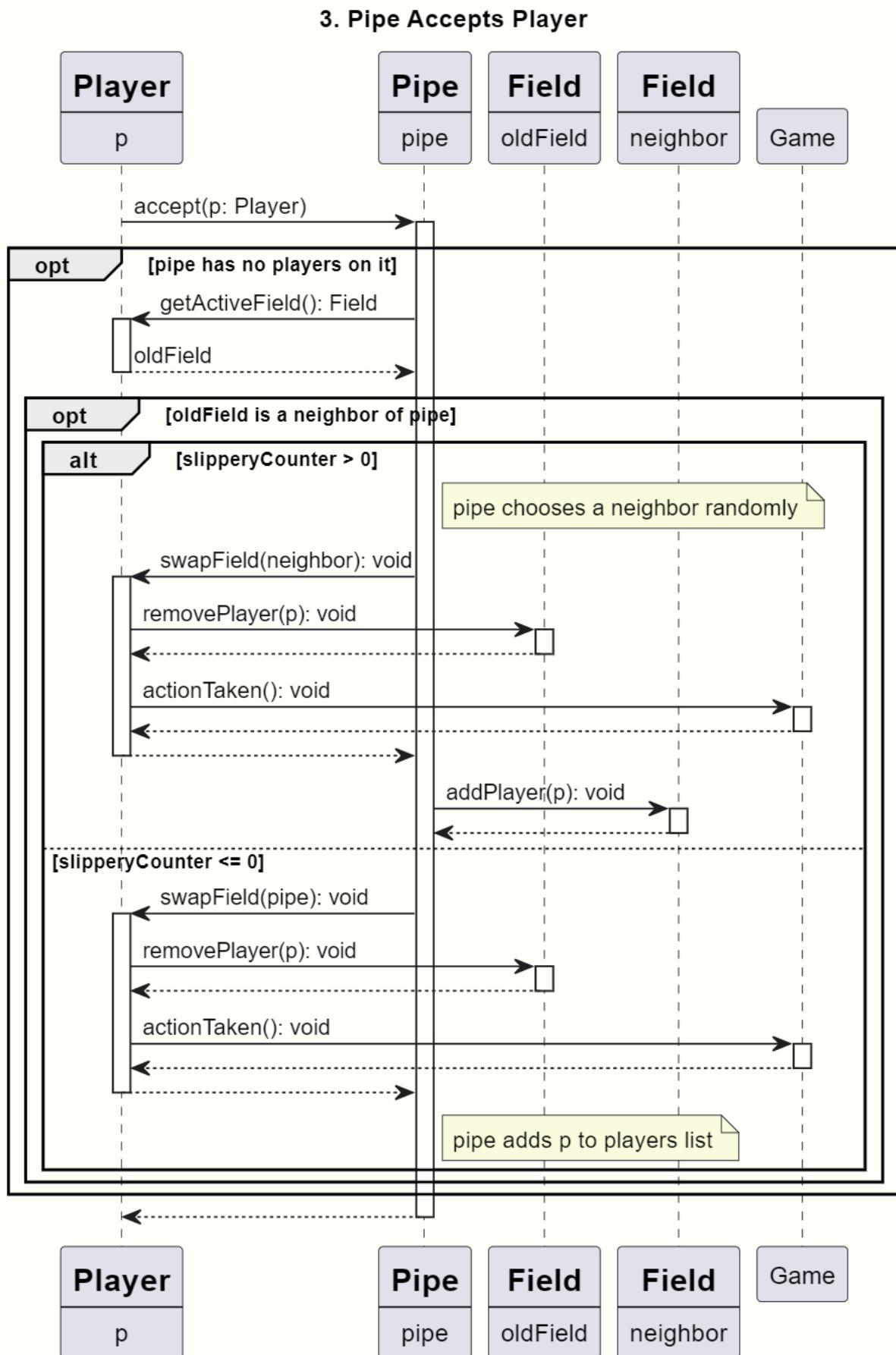
- void makeSlippery(): Meghívja az activePlayer tagváltozón a makeSlippery() metódust.
- void makeSticky(): Meghívja az activePlayer tagváltozón a makeSticky() metódust.
- void movePipe(p, oldEnd, newEnd): A metódus most már akkor is engedi az átmozgatást newEnd = null esetén, ha a p Pipe másik vége sincs sehol csatlakoztatva.

## 7.0.3 Szekvencia-diagramok

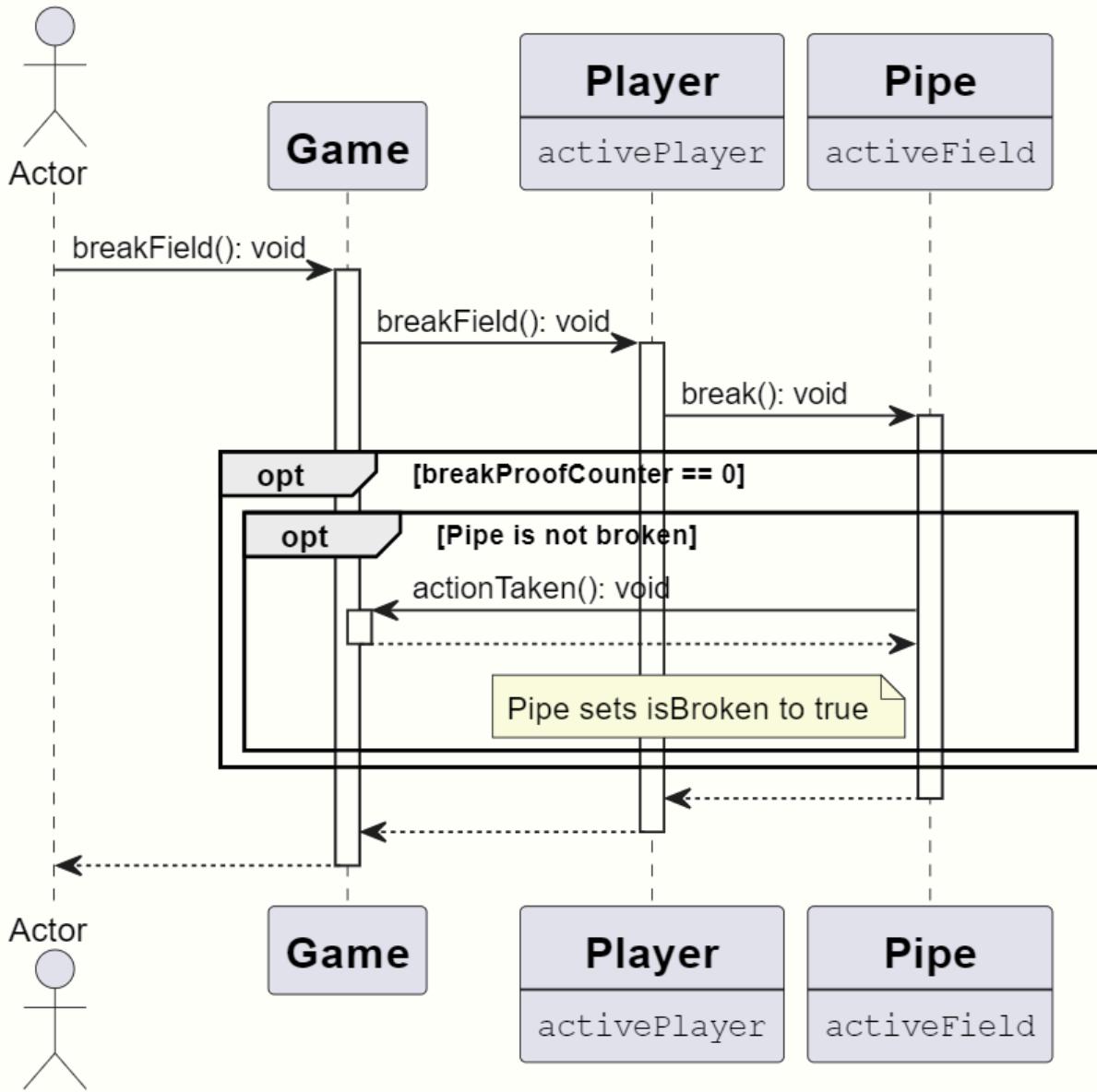
### 2. Player Moves



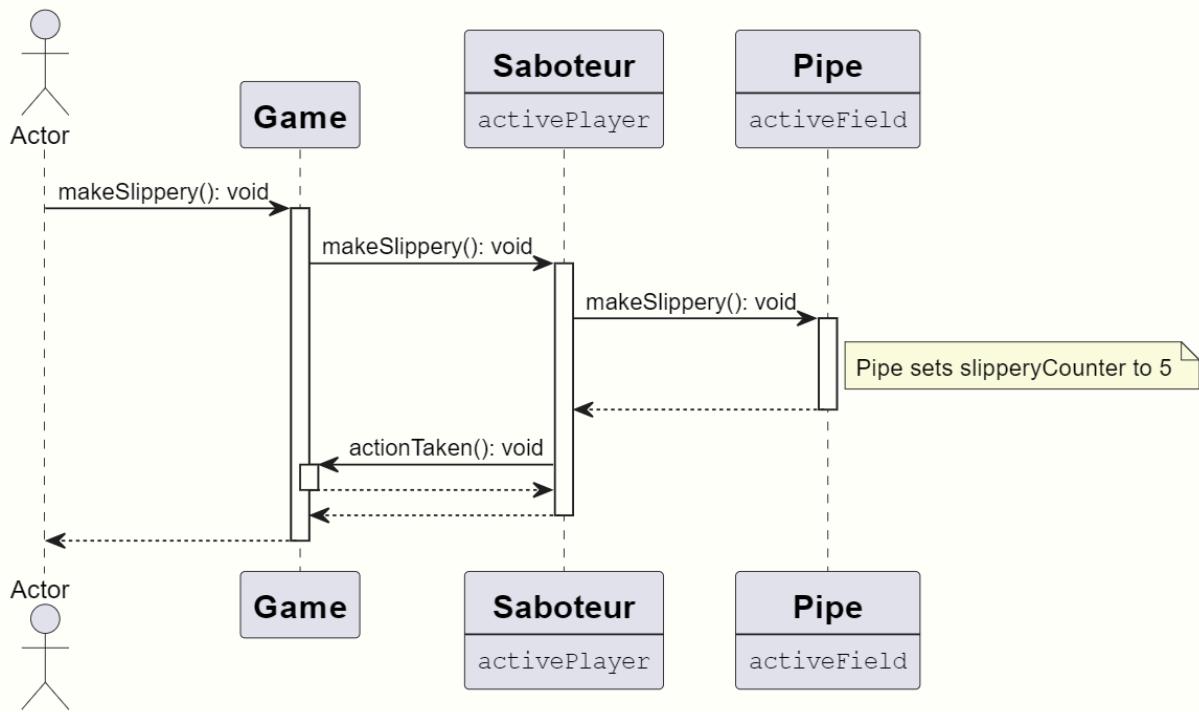
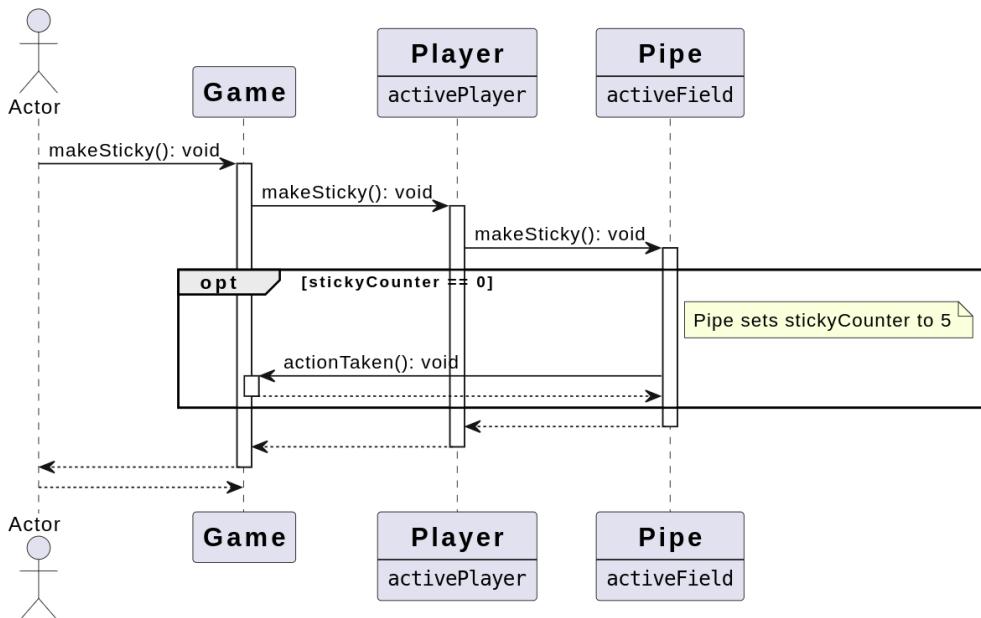
### 3. Pipe Accepts Player



## 9. Player Breaks Pipe (Saboteur Breaks Pipe helyett)

**9. Player Breaks Pipe**

## 22. Saboteur Makes Pipe Slippery

**22. Saboteur Makes Pipe Slippery****23. Player Makes Pipe Sticky****23. Player Makes Pipe Sticky****7.1 Prototípus interface-definíciója****7.1.1 Az interfész általános leírása**

- A prototípus bemeneti és kimeneti felülete egyaránt karakteres alapon működik. Bemenetnek parancsok sorozatát várja, egymástól 'n' karakterekkel elválasztva (tehát 1-1 utasítás külön sorba kell, hogy kerüljön). A program tud fájlból vagy terminálról olvasni*

*illetve fájlba vagy terminálba írni. Ezek között ugyancsak parancsokkal tudunk váltani (7.1.2 fejezet). A prototípus indításkor minden karakteres interfész tekinti a bemenetnek és a kimenetnek is. A prototípus minden releváns osztálya megvalósítja a Serializable interfész, és a [Serialization](#) segítségével lesz elmentve a pályakép.*

### 7.1.2 Bemeneti nyelv

Egy parancs a szintaktikája a következőképp épül fel:

<parancs> <?opcion1, ?opcion2, ?...>

A parancsok listája:

**exit**

**Leírás:** Kilép a programból.

**Opciók:** -

**play**

**Leírás:** Elindít egy új játékot (a korábban betöltött pályán, ha a parancs kiadása előtt nem lett betöltve pálya, a beépítetten fog elindulni).

**Opciók:** -

**end**

**Leírás:** Befejezi a jelenlegi játékot.

**Opciók:** -

**load**

**Leírás:** Egy adott fájlból betölti a játék egy állapotát, a kimeneti nyelven kiírja a beolvasott adatokat, majd elindítja azt. A fájl tartalmának meg kell felelnie a 7.1.2 fejezetben leírtaknak.

**Opciók:**

- **-filename / -f:** Az állapotleíró fájl elérési útja (a futtatható állományhoz képest illetve megadható a teljes elérési út is).

**endTurn**

**Leírás:** Befejezi a jelenlegi kört.

**Opciók:** -

**breakField**

**Leírás:** A jelenlegi aktív játékos mezőjét tönkreteszzi (amennyiben ez lehetséges ezen a mezőn ennek a játékosnak ebben a körben).

**Opciók:** -

**repairField**

**Leírás:** A jelenlegi aktív játékos mezőjét megjavítja (amennyiben ez lehetséges ezen a mezőn ennek a játékosnak ebben a körben).

**Opciók:** -

**place**

**Leírás:** Lerakat a jelenlegi aktív játékossal egy pumpát annak mezőjére (amennyiben ez lehetséges ezen a mezőn ennek a játékosnak ebben a körben).

**Opciók:** -

**pickup**

**Leírás:** Felvetet egy pumpát a jelenlegi aktív játékossal annak mezőjéről (amennyiben ez lehetséges ezen a mezőn ennek a játékosnak ebben a körben).

**Opciók:** -

**changePumpDirection**

**Leírás:** Megváltoztatja a pumpa irányát, amin az aktív játékos éppen áll. (amennyiben a játékos pumpán áll és ebben a körben ez lehetséges).

**Opciók:**

- **-inPipe / -ip:** Az új bemeneti cső neve
- **-outPipe / -op:** Az új kimeneti cső neve

**movePipe**

**Leírás:** Elmozgatja az opciót megadott cső egyik végét egy másik helyre. Amennyiben az új helyet nem adjuk meg ez azt jelenti, hogy nem csatlakoztatjuk azt a véget sehova.

**Megjegyzés:** A módosult követelmények értelmében egy csőnek minden vége "lóghat", azaz lehetséges, hogy semelyik vége sincs sehova csatlakoztatva. Ezt úgy lehet elérni, hogy kétszer is ezt a parancsot adjuk ki ugyanazon cső ("pipe") 1-1 végrére ("oldEnd"), mindenkor a "newEnd" opció nélkül.

**Opciók:**

- **-pipe / -p:** A cső neve, amelyiken végre akarjuk hajtani a műveletet
- **-oldEnd / -oe:** A csőnek az a vége, amit el akarunk mozdítani máshova
- **-newEnd / -ne:** A csőnek az elmozdított véget hova kössük (opcionális)

**makeSlippery**

**Leírás:** Csúszóssá változtatja a jelenlegi aktív játékos mezőjét (amennyiben ez lehetséges).

**Opciók:** -

**makeSticky**

**Leírás:** Ragadóssá változtatja a jelenlegi aktív játékos mezőjét (amennyiben ez lehetséges).

**Opciók:** -

**move**

**Leírás:** Elmozgatja az aktív játékos karakterét az opciót megadott mezőre (amennyiben ez lehetséges).

**Opciók:** **-field / -f:** A mező neve, amelyikre mozgatjuk a karaktert.

**random**

**Leírás:** Ki-be kapcsolja a program véletlenszerűségét. Kikapcsolt állapot esetén a program minden véletlenszerű döntésről megkérdezi a felhasználót.

**Opciók:** -

**input**

**Leírás:** Megadhatunk egy fájlt, amiből egy tesztforgatókönyvet beolvas és lefuttat.

**Opciók:**

- **-filename / -f:** A fájl neve

**output**

**Leírás:** Megadhatunk egy fájlt, amibe az innentől kezdve érkező parancsok kimenetét írja a program. Amennyiben nem adunk meg paramétert, a terminálba irányul a kimenet.

**Opciók:** -

- **-filename / -f:** A fájl neve

*A pályaképet megadó fájlok nyelvtana a Game osztály egy példánya JSON formátumba konvertálva, a 7.1.3 fejezetben leírtaknak megfelelően.*

*Megjegyzés: a move, makeSticky, makeSlippery, movePipe, changePumpDirection, pickup, place, repairField, breakField és endTurn parancsok csak azután értelmesek, hogy*

*elindították a játékot (play parancs). Ezért, ha a játék indítása előtt adjuk ki őket, nem történik semmi.*

### 7.1.3 Kimeneti nyelv

A kimeneti nyelv formátuma JSON alapú, viszont fontos logikai különbségek vannak:

- Egy osztály neve "<osztálynév>#<ID>" formátumú, ahol az ID egy az osztályhoz tartozó egyedi azonosító. A prototípus program ennek segítségével tudja azonosítani az objektumokat.
- Ha egy osztálynak tagváltozója egy primitív típus, az a következőképp jelenik meg:

```
"<változó neve>": <érték>
```

- Ahol az érték lehet: true, false, szám, "szöveg".
- Ha egy osztálynak tagváltozója egy lista, az a következőképp jelenik meg:

```
"<lista neve>":
[
    "0:<elem0név>": <...listaElem0 kifejtése...>,
    "1@[Reference)": <listaElem1osztály#ID>
]
```

- A lista első eleme azt mutatja be, ha az elem egy olyan objektum, amivel a parser még nem találkozott. Ekkor ezt itt a listán belül kifejti, amíg olyan elemekig leér, amik vagy primitív típusok vagy referenciaiak.
- A lista második eleme azt mutatja be, ha az elem egy olyan objektum, amivel a parser már korábban találkozott, ezért itt csak referenciaiként megnevezzi. Ez azt jelenti, hogy ha kíváncsiak vagyunk arra, hogy néz ki ez az objektum, meg kell keresni az első előfordulását.
- Ha egy osztály tagváltozója egy másik osztály egy példánya, kétféleképp jelenhet meg:

```
"<tagváltozó neve>:<tagváltozó osztálya>#<ID>":
{...<tagváltozó kifejtése>...}

"<tagváltozó neve>@[Reference)": <tagváltozó
osztálya>#<ID>
```

- Az első esetben egy olyan objektum a tagváltozó, amivel a parser még korábban nem találkozott, ezért azt itt kifejti, egészen addig, amíg csak olyan elemeket talál, amik vagy primitív típusúak, vagy csak referenciaiák.
- A második esetben egy olyan objektum a tagváltozó, amivel a parser már találkozott, ezért itt csak referenciaként megnevezi. Ez azt jelenti, hogy ha kíváncsiak vagyunk arra, hogy néz ki ez az objektum, meg kell keresni az első előfordulását.
  - Egy osztálynak ha olyan tagváltozója van, ami üres (null) ezt a parser nem fogja kiírni.

Példa a kimeneti nyelvre:

Megjegyzés: A <...>-al jelölt részeken a betűményet miatt nem fért ki a maradék rész, de ez nem azt jelenti, hogy itt végtelen sokáig menne a körkörös dependenciák miatt a fájl, a fentiekben leírtak miatt ugyanis ez nem lehetséges.

```
"Game#0" :
{
  "players" :
  [
    "0:Mechanic#1" :
    {
      "activeField:Cistern#2" :
      {
        "neighbors" :
        [
          "0:Pipe#3" :
          {
            "neighbors" :
            [
              "0@[Reference]" : Cistern#2,
              "1:Pump#4" :
              {
                "neighbors" :
                [
                  "[<...>]",
                  "players" :
                  [
                    "0:Saboteur#5" :
                    {<...>}
                  ],
                  "hasStoredWater" :
                  false,
                  "isBroken" : false
                }
              ],
              "players" :
              [],
              "oldWaterState" : false,
            ]
          }
        ]
      }
    }
  ]
}
```

```

        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
],
"players":
[
    "0@[Reference)": Mechanic#1
]
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 0,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}
```

## Összes részletes use-case

Use-case neve	play
---------------	------

<b>Rövid leírás</b>	Elindítja a játékot.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	A felhasználó elindítja a játékot a play parancssal.

<b>Use-case neve</b>	end
<b>Rövid leírás</b>	Befejezi a jelenlegi játékot. Meghívódik az end() függvény.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	A játék megáll, egy utolsó kiíratást végez a kimenetre.

<b>Use-case neve</b>	load
<b>Rövid leírás</b>	A játék egy érvényes állapotát lehet beolvasni.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	Az opcióként megadott fájlból beolvassa a játék egy érvényes állapotát (pl érvényes beolvasás az output parancsban megadott file beolvasása).

<b>Use-case neve</b>	endTurn
<b>Rövid leírás</b>	Az aktív játékos körét be szeretné fejezni. Meghívódik az endTurn() függvény.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	A parancs következményeként az aktív játékos megváltozik, a soron következő játékos lesz az aktív játékos, az egyes komponensek elvégzik a kör végi dolgaikat.

<b>Use-case neve</b>	breakField
<b>Rövid leírás</b>	Egy játék futása során, az aktív játékos az alatta lévő mezőt próbálja tönkretenni. Meghívódik a breakField() függvény.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	A parancs beírása után, az aktív játékos alatt lévő mező eltörhet, ha ez lehetséges, és ha ez megtörtént, a körben hátralévő akciók száma csökken 1-el.

<b>Use-case neve</b>	repairField
<b>Rövid leírás</b>	Egy játék futása során, az aktív játékos az alatta lévő mezőt meg próbálja javítani. Meghívódik a repairField() függvény.
<b>Aktorok</b>	Player

<b>Forgatókönyv</b>	A parancs beírása után, amennyiben sikerült megjavítani a mezőt, a Mechanic aktív játékos alatt lévő cső állapotán megfigyelhető, hogy az isBroken = false; A breakProofCounter értéke pedig egy 0-tól különböző szám lesz illetve a körben hátralévő akciók száma csökken 1-el.
---------------------	--

<b>Use-case neve</b>	place
<b>Rövid leírás</b>	Egy játék futása során a Mechanic karakter a nála lévő pumpát le próbálja tenni. Meghívódik a place() függvény.
<b>Aktorok</b>	Mechanic
<b>Forgatókönyv</b>	Amennyiben van a szerelőnél pumpa, és eljutott egy csőre, akkor a place parancs után a játékos alatti terület megváltozik egy újonnan létrehozott pumpává. A pumpára két új cső fog a csatlakozni, ami összeköti azzal a két pumpával, ami az eredeti cső két végén volt.

<b>Use-case neve</b>	pickup
<b>Rövid leírás</b>	Egy játék futása közben az aktív játékosra meghívódik a pickup() függvény.
<b>Aktorok</b>	Player
<b>Forgatókönyv</b>	A pickup parancs következményében, ha az aktív játékos Mechanic volt, akinél nincs pumpa és egy olyan ciszternán helyezkedett el, ahol van, akkor a kimenetén észrevehető hogy az adott ciszternán, nem lesz pumpa, illetve hogy az aktív játékos tarsolyában lesz egy pumpa.

<b>Use-case neve</b>	changePumpDirection
<b>Rövid leírás</b>	Egy játék futása alatt az aktív játékosra meghívódik a changePumpDirection(pipe in, pipe out) függvény
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	Egy Mechanic játékos körében, egy pumpán állva a changePumpDirection parancs meghívása után észre lehet venni hogy az adott pumpa aktiv kimenete és aktiv bemenete meg fog felelni az opcióként átadott elemekkel, akkor hogy ha ezek az elemek valamelyik szomszédos csövet képviselnek. Bármelyik opciót üresen hagyva a megfelelő mező értéke null lesz (ezzel azt jelölve, hogy a pumpának a kiválasztott végét nem szeretnénk sehol kötni).

<b>Use-case neve</b>	movePipe
<b>Rövid leírás</b>	Az aktív játékoson meghívódik a movePipe(Pipe p, Field oldEnd, Field newEnd).
<b>Aktorok</b>	Felhasználó.

<b>Forgatókönyv</b>	A parancs következtében észrevesszük hogy az a cső, amit első paraméterben megadtunk, már nem illeszkedik az oldEnd opció által specifikált mezőre, illetve ez a mező sem szomszédos a csővel, hanem most már a newEnd opció által specifikált mező jelenik meg helyette, illetve ennek a szomszédsági listájában is megtalálhatjuk a p Pipe-ot.
---------------------	--

<b>Use-case neve</b>	makeSlippery
<b>Rövid leírás</b>	Meghívja a játékoson a makeSlippery() függvényt
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	Ha az aktív játékos egy szabotőr és az alatta lévő mező egy cső, ekkor a makeSlippery parancs során az adott mező slipperyCounter értéke 0-nál nagyobb lesz illetve a stickyCounter értéke 0 lesz.

<b>Use-case neve</b>	makeSticky
<b>Rövid leírás</b>	Meghívja a játékoson a makeSticky() függvényt.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	Ha az aktív játékos alatt lévő mező egy cső, ekkor a makeSticky parancs során az adott mező slipperyCounter értéke 0 lesz illetve a stickyCounter értéke 0-nál nagyobb szám.

<b>Use-case neve</b>	random
<b>Rövid leírás</b>	Ki és bekapcsolja a játék determinisztikus viselkedését.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	A parancs meghívásakor a további parancsot, mint például a makeSlippery, makeSticky vagy repairField esetén, további információt kell adni, hogy hány körre szeretnénk hogy fennmaradjon a definiált viselkedés.

<b>Use-case neve</b>	input
<b>Rövid leírás</b>	Teszt forgatókönyveket lehet beolvasni.
<b>Aktorok</b>	Felhasználó
<b>Forgatókönyv</b>	Érvényes parancsokat listázó fájlt kap meg opcióként a parancs, amit sorban elvégez.

<b>Use-case neve</b>	output
<b>Rövid leírás</b>	Azt adhatjuk meg hogy hova szeretnénk tovább a kiíratást végezni.
<b>Aktorok</b>	Felhasználó

<b>Forgatókönyv</b>	A paraméterben megkapott fájlba fogja a további érvényes állapot kiírásokat végezni (inne később akár be lehet olvasni a load parancsal). Ha a fájl nem létezik, akkor létrehoz a paraméterrel azonos nevűt. Ha nincs specifikálva opción, akkor konzolra jeleníti meg az információt.
---------------------	--

## 7.2 Tesztelési terv

<b>Teszt-eset neve</b>	Az aktív játékos sikeresen mozog
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy mozgató parancsot, a játékos rálép a célként megadott mezőre és veszít egy akciót pontot. A mozgás azért sikeres, mert az adott mezőre lehet lépni (a célpont szomszédos és cső esetén nem áll rajta más) és a játékosnak van még akciót pontja ebben a körben illetve el tud lépni a mezőjéről (nincs beleragadva).
<b>Teszt célja</b>	Tesztelt funkció: aktív játékos mozgatása, az ehhez szükséges követelmények ellenőrzése. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: move.

<b>Teszt-eset neve</b>	Az aktív játékos csúszós csőre lép
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy mozgató parancsot, a játékos rálép a célként megadott mezőre és veszít egy akciót pontot. Mivel a célként választott cső csúszós, a játékos a cső egy véletlenszerűen választott szomszédjára kerül. A mozgás sikeres, mert az adott csövön nem állt senki, a szomszédai pedig nem lehetnek csövek, tehát nem fordulhat elő, hogy nem tud rájuk lépni a játékos.
<b>Teszt célja</b>	Tesztelt funkciók: aktív játékos mozgatása, az ehhez szükséges követelmények ellenőrzése, csúszós cső viselkedése. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: move, random.

<b>Teszt-eset neve</b>	Az aktív játékos nem tud lépni a kívánt mezőre
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy mozgató parancsot, a játékos megpróbál a célként megadott mezőre lépni, de nem tud a következő okok valamelyike miatt: a célként megadott mező egy cső, amin már állnak, a játékos ragadós csövön áll, a játékosnak nincs több akciót pontja, vagy a célként megadott mező nem szomszédos az aktuális tartózkodási helygel.
<b>Teszt célja</b>	Tesztelt funkciók: aktív játékos mozgatása, ragadós cső viselkedése, foglalt cső viselkedése, akciót pontok kezelése, szomszédosság ellenőrzése. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: move.

<b>Teszt-eset neve</b>	A játékos sikeresen kilyukaszt egy csövet
------------------------	---

<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet kilyukasztó parancsot, melynek következtében a cső, amin a játékos éppen áll, kilyukad. A lyukasztás azért sikeres, mert a cső eddig nem volt eltörve, valamint éppen törhető állapotban van (mert nem volt foltozva, vagy ha volt, akkor már olyan régen, hogy újra lyukasztható lett).
<b>Teszt célja</b>	Tesztelt funkciók: játékos csövet lyukaszt. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: breakField.

<b>Teszt-eset neve</b>	A játékos nem tud kilyukasztani egy csövet
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet kilyukasztó parancsot, de a cső, amelyen a játékos éppen áll, mégsem lyukad ki a következő okok valamelyike miatt: a cső eleve lyukas volt, vagy a cső nemrég volt foltozva és még nem lyukasztható ki újra.
<b>Teszt célja</b>	Tesztelt funkciók: játékos csövet lyukaszt, lyukasság ellenőrzése, javítási utáni lyukasztás gátolásának ellenőrzése. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: breakField.

<b>Teszt-eset neve</b>	Szerelő sikeresen megjavít egy mezőt
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy javító parancsot, melynek következtében a mező, amin a szerelő áll, megjavul és a játékos veszít egy akciót pontot. A javítás azért volt sikeres, mert a mező, amin a játékos áll, cső vagy pumpa és valóban el volt romolva.
<b>Teszt célja</b>	Tesztelt funkciók: szerelő javít. Tesztelt osztályok: Game, Field, Player, Mechanic. Tesztelt parancsok: repairField.

<b>Teszt-eset neve</b>	Szerelő nem tud megjavítani egy mezőt
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy javító parancsot, de a mező, amin a játékos áll, nem javul meg a következő okok valamelyike miatt: a mező nem cső vagy pumpa (tehát eleve nem lehetett elromolva), vagy a mező ugyan cső vagy pumpa, de nem volt elromolva.
<b>Teszt célja</b>	Tesztelt funkciók: szerelő javít, pumpa működöképességének ellenőrzése, cső lyukasságának ellenőrzése. Tesztelt osztályok: Game, Field, Player, Mechanic. Tesztelt parancsok: repairField.

<b>Teszt-eset neve</b>	Szabotőr csúszóssá tesz egy csövet
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csúszóssá tevő parancsot, melynek következtében a cső, amelyen a szabotőr áll, a következő öt körben csúszós lesz. A művelet mindenkorábban sikeres (feltéve, hogy a játékosnak van akciót pontja),

	hiszen nem fordulhat elő, hogy a cső már csúszós volt, mert akkor nem állhatna rajta játékos.
<b>Teszt célja</b>	Teszttel funkciók: szabotör csúszóssá tesz egy csövet. Teszteld osztályok: Game, Player, Saboteur, Field, Pipe. Tesztelt parancsok: makeSlippery.

<b>Teszt-eset neve</b>	Játékos ragadóssá tesz egy csövet
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy ragadóssá tevő parancsot, melynek következetében a cső, amelyen a játékos áll, a következőöt körben ragadós lesz. Amennyiben a cső eleve ragadós volt, a parancs egyszerűen visszaállítja a ragadósság időtartamát ötkörre, a művelet ettől még sikeres (feltéve, hogy a játékosnak van akciót pontja).
<b>Teszt célja</b>	Teszttel funkciók: játékos ragadóssá tesz egy csövet. Teszteld osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: makeSticky.

<b>Teszt-eset neve</b>	Játékos befejezi a körét
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy kör vége parancsot, melynek következetében az aktív játékos a következő játékos lesz, valamint a játék átáll az egy körrrel későbbi állapotba (tehát végbemennek a változások, amik a kört befejező játékos körében történtek).
<b>Teszt célja</b>	Teszttel funkciók: játékos befejezi a körét. Teszteld osztályok: Game, Player, Timer. Tesztelt parancsok: endTurn.

<b>Teszt-eset neve</b>	Szerelő sikeresen letesz egy új pumpát
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát letesz parancsot, melynek következetében a cső, amin a szerelő áll, megfeleződik és a két fél közé beékelődik az új pumpa. A művelet azért sikeres, mert a szerelő valóban csövön áll, van nála pumpa és van még akciót pontja.
<b>Teszt célja</b>	Teszttel funkciók: szerelő pumpát tesz le. Teszteld osztályok: Game, Player, Mechanic, Field, Pipe, Pump. Tesztelt parancsok: place.

<b>Teszt-eset neve</b>	Szerelő nem tud pumpát letenni
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát letesz parancsot, de nem sikerül a művelet a következő okok valamelyike miatt: a szerelő nem csövön áll, a szerelőnél nincs pumpa, a játékosnak nincs több akciót pontja.

<b>Teszt célja</b>	Teszttel funkciók: szerelő pumpát tesz le, játékosnál van-e pumpa ellenőrzés. Tesztelt osztályok: Game, Player, Mechanic, Field, Pipe, Pump. Tesztelt parancsok: place.
--------------------	---

<b>Teszt-eset neve</b>	Szerelő sikeresen felvesz egy pumpát
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát felvesz parancsot, melynek következetében a szerelőhöz kerül egy pumpa. A művelet azért volt sikeres, mert a szerelő ciszternán áll, a ciszternán van pumpa és a játékosnak van még akciópontja.
<b>Teszt célja</b>	Teszttel funkciók: szerelő pumpát felvesz. Tesztelt osztályok: Game, Player, Mechanic, Field, Cistern, Pump. Tesztelt parancsok: pickup.

<b>Teszt-eset neve</b>	Szerelő nem tud pumpát felvenni
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát felvesz parancsot, de a szerelőhöz nem kerül pumpa a következő okok valamelyike miatt: a szerelő nem ciszternán áll, a szerelő olyan ciszternán áll amin nincs pumpa, a játékosnak nincs több akciópontja, a játékosnál már van pumpa.
<b>Teszt célja</b>	Teszttel funkciók: szerelő pumpát felvesz, ciszternán van-e pumpa ellenőrzés, szerelőnél van-e pumpa ellenőrzés. Teszteld osztályok: Game, Player, Mechanic, Field, Cistern, Pump. Tesztelt parancsok: pickup.

<b>Teszt-eset neve</b>	Játékos sikeresen átcsatlakoztat egy csővéget
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet átcsatlakoztat parancsot, melynek következetében egy cső vége más mezőhöz fog kapcsolódni. A művelet azért sikeres, mert az új végként megadott mező nem egyezik meg az eddigi véggel, valamint a játékosnak van még akciópontja.
<b>Teszt célja</b>	Teszttel funkciók: játékos csövet átköt. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: movePipe.

<b>Teszt-eset neve</b>	Játékos nem tud átkötni egy csővéget
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet átcsatlakoztat parancsot, de a cső vége nem fog más mezőhöz kapcsolódni a következő okok valamelyike miatt: a megadott új vég megegyezik az eddigivel, vagy a játékosnak nincs több akciópontja.

<b>Teszt célja</b>	Teszttelt funkciók: játékos csövet átköt, új vég = régi vég ellenőrzése. Tesztelt osztályok: Game, Player, Field, Pipe. Tesztelt parancsok: movePipe.
--------------------	---

<b>Teszt-eset neve</b>	Játékos sikeresen átállítja egy pumpa be- és kimenetét
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát átállít parancsot, melynek következetében a pumpa, amelyen a játékos áll, más bemenetről fogad és más kimenetre küld vizet, mint eddig. A művelet azért sikeres, mert a játékos valóban pumpán áll, az új be- és kimenet szomszéda a pumpának, az új be- és kimenet nem ugyanaz a cső és a játékosnak van még akciót pontja.
<b>Teszt célja</b>	Teszttelt funkciók: játékos pumpát átállít. Tesztelt osztályok: Game, Player, Field, Pump, Pipe. Tesztelt parancsok: changePumpDirection.

<b>Teszt-eset neve</b>	Játékos nem tudja átállítani egy pumpa be- és kimenetét
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát átállít parancsot, de a művelet sikertelen a következő okok valamelyike miatt: a játékos nem pumpán áll, a megadott új be- és kimenet valamelyike (vagy mindenkettő) nem szomszédos a pumpával, amin a játékos áll, a megadott új be- és kimenet ugyanaz a cső, a játékosnak nincs több akciót pontja.
<b>Teszt célja</b>	Teszttelt funkciók: pumpát átállít, szomszédos-e a megadott mező ellenőrzés, megegyezik-e a két megadott mező ellenőrzés. Teszteld osztályok: Game, Player, Field, Pump, Pipe. Tesztelt parancsok: changePumpDirection.

<b>Teszt-eset neve</b>	(összetett) Lépés + csúszás + pumpa átállítás
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: léptető, pumpát átállító. A léptető parancs következetében a játékos rálép a célként megadott mezőre, ami egy csúszós cső, ezért átcsúszik egy pumpára. A pumpára érve a második parancccsal átállítja a pumpa irányát, feltéve, hogy a megadott új be- és kimenet érvényes és a játékosnak van még akciót pontja.
<b>Teszt célja</b>	Teszttelt funkciók: pumpát átállít, szomszédos-e a megadott mező ellenőrzés, megegyezik-e a két megadott mező ellenőrzés, csúszós cső viselkedése, játékos lépése, random választás (csúszós csőről hova menjen). Tesztelt osztályok: Game, Field, Pipe, Pump, Player. Tesztelt parancsok: move, random (csúszásnál), changePumpDirection.

<b>Teszt-eset neve</b>	(összetett) Játékos lyukaszt, csúszóssá tesz majd ellép
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: csövet kilyukaszt, csúszóssá tesz, lép. Az első parancs hatására a játékos kilyukasztja a csövet, amin

	áll (feltéve, hogy csövön áll és az lyukasztható állapotban van), a második következtében csúszóssá teszi a csövet, amin áll (ezt mindenkihez megteheti, ha van akciótömb), végül ellép egy megadott pályaelemre (ezt biztosan megteheti, ha van elég akciótömb, hiszen csőnek nem lehet cső a szomszédja).
<b>Teszt célja</b>	Tesztelt funkciók: csövet lyukaszt, lyukasztható állapot ellenőrzése, csövet csúszóssá tesz, lépés. Tesztelt osztályok: Game, Field, Pipe, Player. Tesztelt parancsok: breakField, makeSlippery, move.

<b>Teszt-eset neve</b>	(összetett) Szerelő felvesz egy pumpát, lép, majd leteszi
<b>Rövid leírás</b>	A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: pumpát felvesz, lép, pumpát letesz. Az első hatására a szerelő játékos felvesz egy pumpát a cisternáról, amin áll (ha azon áll és ott van pumpa), a második parancs hatására ellép a megadott csőre (feltéve, hogy ott nem áll más, cisternának csak cső lehet a szomszédja), végül leteszi a pumpáját a csőre, amin áll (ha maradt még akciótömb).
<b>Teszt célja</b>	Tesztelt funkciók: pumpát felvesz, van-e pumpa a cisternán ellenőrzés, van-e pumpa a játékosnál ellenőrzés, lépés, csövön állnak-e ellenőrzés, pumpát letesz. Tesztelt osztályok: Game, Field, Pipe, Pump, Cistern, Player, Mechanic. Tesztelt parancsok: pickup, move, place.

A kimenet minden teszt esetén a megadott fájlba, vagy ennek hiján a szabványos kimenetre írodik. A kimenetet összehasonlíthatunk az elvárt kimenettel és amennyiben egyezik a kettő, a teszt sikeres. Bővebben lejjebb.

### 7.3 Tesztelést támogató segéd- és fordítóprogramok specifikálása

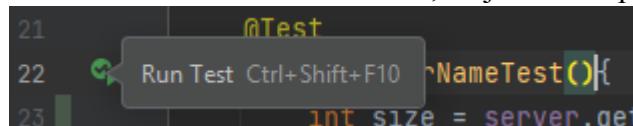
#### Segédelszközök: IntelliJ alatt JUnit 5.8.1 könyvtár

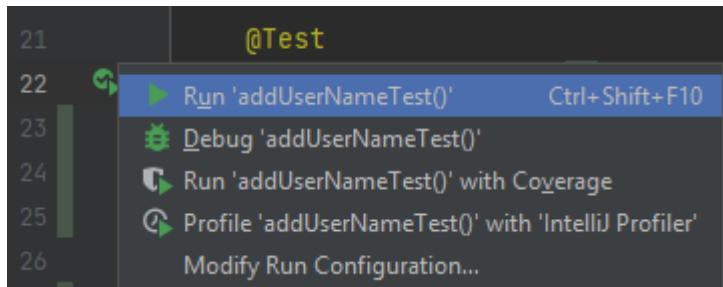
A tesztelést JUnit segítségével fogjuk végezni. Ehhez az IntelliJ projektbe importáljuk a JUnit5.8.1 külső könyvtárat. A projekt tartalmazni fog egy Test nevű osztályt, melyben a JUnit tesztek lesznek definiálva.

#### Tesztek indítása:

Egyenként:

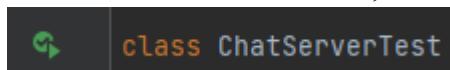
IntelliJ-ben a tesztesetek egyenként futtathatóak a függvények mellett kis zöld ikonra kattintással, majd a Run opció kiválasztásával.





Közösen:

Az összes teszt együttes futtatásához a Test osztálytól balra lévő kis zöld ikont kell használni, ez esetben lefut sorban az összes teszt.



### Tesztek sikerességének vizsgálata:

A tesztek elején a System.in-t átállítjuk egy ByteArrayInputStream-re, melyben megadunk egy input parancssal egy fájlt, amiben a teszt parancsai vannak sorban. Ezek között az output parancs is szerepel az elején, mely beállít egy kimeneti fájlt. Ezáltal minden teszthez lesz egy kimeneti fájl, melynek a tartalmát összehasonlítjuk az elvárt tartalommal, és ha ez egyezik, akkor a teszt sikeresen lefutott.

## Napló

Kezdet	Időtartam	Résztvevők	Leírás
2023.04.21. 15:30	1.5 óra	Varga Mali Garai Zavadil	Értekezlet. Feladatok felosztása: 7.4 - Garai 7.3 - Völgyesi 7.1 - Zavadil 7.0 - Mali 7.2 - Varga
2023.04.21. 15:00	3 óra	Zavadil	7.1 elkészítése
2023.04.21. 18:00	3 óra	Mali	7.0 elkészítése
2023.04.22 8:00	3 óra	Varga	Use-case-ek megírása
2023.04.22 21:00	3 óra	Völgyesi	Teszt tervezek megírása
2023.04.23 11:00	1 óra	Völgyesi	Teszt tervezek kiegészítése összetett tesztekkel
2023.04.23. 10:30	1.5 óra	Garai	7.3 átnézése, 7.4 megírása
2023.04.23. 20:00	1 óra	Zavadil	Kék szövegek kiszedése, nyelvtani és logikai ellenőrzés, fedlap merge, beadás online

## 8. Részletes tervezetek

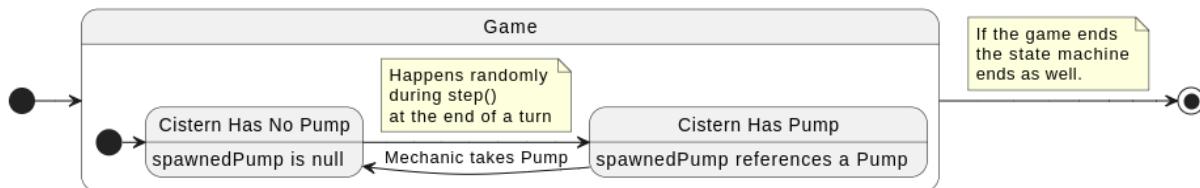
### 8.1 Osztályok és metódusok tervezetek

A prototípus program implementációjában az itt felsorolt osztályok mindegyike megvalósítja a `Serializable` interfést, ennek segítségével a fájlba írás illetve fájlból olvasás megvalósítható (ezt nem jelöljük minden osztálynál külön az átláthatóság érdekében).

#### 8.1.1 Cistern

- **Felelősség**

Amennyiben egy körben folyik bele víz (azaz a bemeneti csövén van víz), pontot kapnak a szerelők. Emellett tárolja, hogy van-e rajta még fel nem vett pumpa, melyet a szerelők felvehetnek. minden körben kis eséllyel generálódik rajta egy pumpa, amennyiben korábban nem volt rajta.



- **Ősosztályok**

*Field*

- **Interfészek**

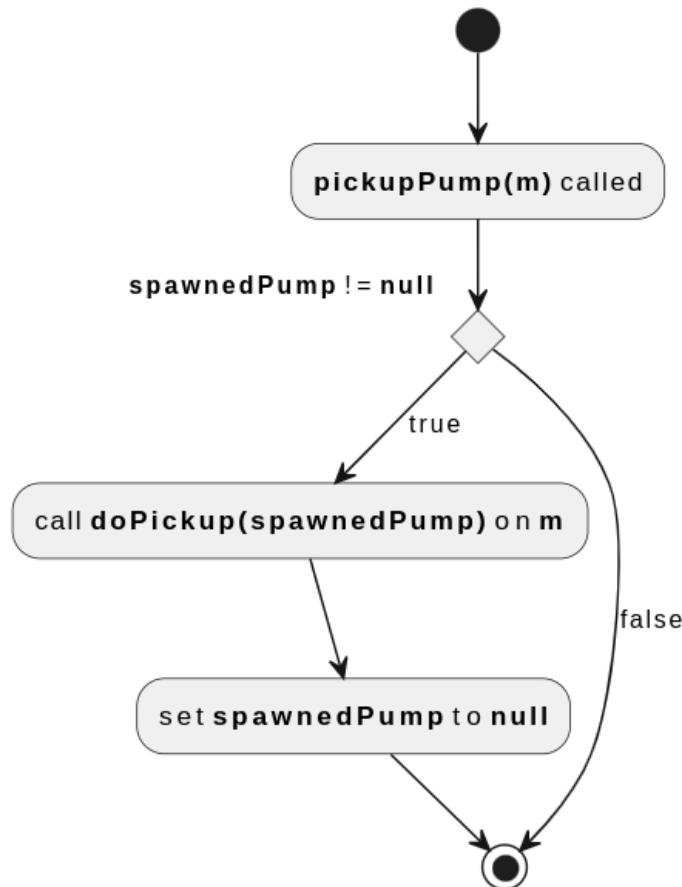
*Periodic*

- **Attribútumok**

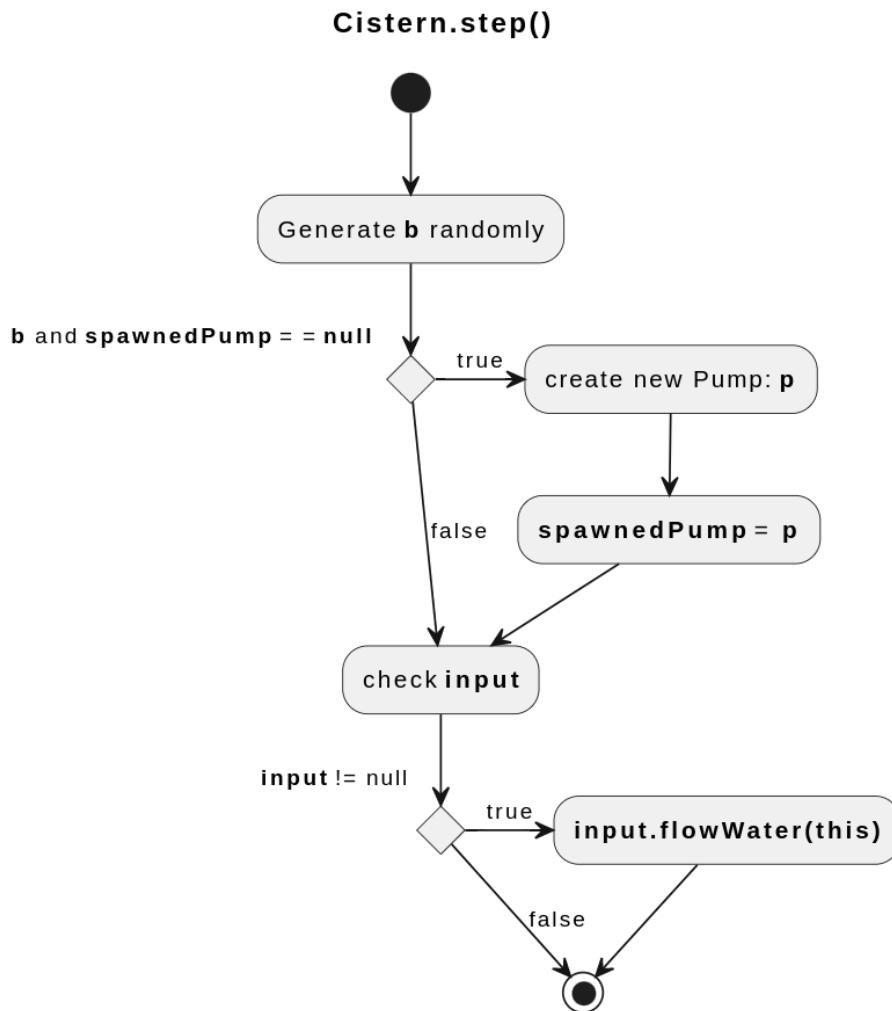
- **-Pipe input:** A ciszterna bemeneti csöve.
- **-Pump spawnedPump:** Ha van a ciszternán pumpa, akkor annak referenciaja. Ezt egy szerelő felveheti. Ha nincs rajta pumpa (nem generálódott vagy már felvették) ez a mező null értékű.

- **Metódusok**

- **+Pump pickupPump(Mechanic m):** Ha van rajta pumpa (azaz `spawnedPump` értéke nem `null`), akkor azt a paraméterül kapott szerelőnek paraméterként átadja a `doPickup(Pump p)` metódousának meghívásával, ezt követően pedig beállítja a `spawnedPump` értékét `null`-ra. Amennyiben nincs rajta pumpa, nem csinál semmit a függvény.

**Cistern.pickupPump()**

- **+void step():** Minden körben, ha a bemeneti csövén van víz, akkor a Game osztályon keresztül ad egy pontot a szerelőknek, a illetve ha nem volt eddig pumpája, kis eséllyel generálódik rajta egy, azaz a `spawnedPump` attribútumának az értéke egy újonnan létrehozott pumpa referenciája lesz.



*Megjegyzés: Az aktivitásdiagramon b egy véletlenszerűen generált változó, melynek értéke igaz vagy hamis.*

- **+void addWater():** Víz kerül a ciszternába, pontot ad a szerelőknek a **Game** osztály **addMechanicPoints()** metódusának meghívásával.
- **+void addNeighbor(Pipe p):** meghívja a szülő (**Field**) osztály **addNeighbor(Field f)** metódusát, paraméterként átadva neki **p-t**, majd az **input** tagváltozóját beállítja **p-re**.
- **+void removeNeighbor(Field f):** meghívja a szülő osztály **removeNeighbor(Field f)** metódusát, paraméterként átadva neki **f-et**, majd az **input** tagváltozóját beállítja **null-ra**.

### 8.1.2 Field

- **Felelősség**

Absztrakt osztály, melynek funkciója, hogy összegyűji azokat az objektumokat, amire a játékosok rá tudnak lépni. Ezen kívül lehetővé teszi azt, hogy a játékosok interakciókat kezdeményezzenek vele: pumpálási irány megváltoztatása, pumpa lerakása, pumpa felvételle, eltörés, megjavítás. Ezen a szinten azonban mindenek hatására nem történik semmi, az egyes leszármazott osztályok viszont saját maguk dönthetik el, hogy ezek közül melyikeket valósítják meg. Fontos funkciója még, hogy eltárolja, milyen más mezőkkel (**Field**) szomszédos. Lehet szomszédot hozzáadni illetve tőle elvenni is.

- Ősosztályok

-

- Interfészek

-

- Attribútumok

- **#List<Player> players:** Számítja, hogy milyen játékosok állnak ezen a mezőn. Ezt az attribútumot az **addPlayer()** és a **removePlayer()** metódusok segítségével lehet manipulálni.
- **#List<Field> neighbors:** Számítja a vele szomszédos mezőket. Ezt az attribútumot az **addNeighbor()** és **removeNeighbor()** metódusokkal lehet manipulálni.

- Metódusok

- **+void addPlayer(Player p):** Felveszi a paraméterként kapott játékost a **players** listába. Ez a játékos most már ezen a mezőn áll.
- **+void removePlayer(Player p):** A paraméterül kapott játékost kiveszi a **players** listából, ezzel azt jelezve hogy a játékos lelépett erről a mezőről.
- **+void accept(Player p):** Amennyiben léphet rá a paraméterül kapott játékos, lépteti őt. Ezen a szinten mindenki léphet rá játékos, tehát meghívja a **swapField()** metódust a paraméterként kapott játékoson és paraméterként önmagát átadja.
- **+void changePumpDirection(Pipe in, Pipe out):** A mezőn nem lehet a pumpálási irányt állítani, a függvény ezen a szinten nem csinál semmit.
- **+void addNeighbor(Field f):** A paraméterül kapott mezőt hozzáveszi a szomszédok listájához (**neighbors**), tehát egy új objektummal szomszédos a jelenlegi mező.
- **+void removeNeighbor(Field f):** a paraméterben kapott mezőt eltávolítja a szomszédok listájából (**neighbors**), tehát ez a szomszéddal megszűnt a kapcsolata.
- **+void placePump(Mechanic m):** Nem tud lerakni a paraméterként kapott játékos erre a mezőre pumpát, a függvény ezen a szinten nem csinál semmit.
- **+void pickupPump(Mechanic m):** Nem tud felvenni erről a mezőről a paraméterként kapott játékos pumpát, a függvény ezen a szinten nem csinál semmit.
- **+void repair():** Ezt a mezőt nem lehet megjavítani, a függvény ezen a szinten nem csinál semmit.
- **+void break():** Ezt a mezőt nem lehet törikretni, a függvény ezen a szinten nem csinál semmit.
- **+void makeSlippery():** A mezőt nem lehet csúszóssé tenni, a függvény ezen a szinten nem csinál semmit.
- **+void makeSticky():** A mezőt nem lehet ragadóssé tenni, a metódus implementációja ebben az osztályban üres.
- **+bool isSticky():** Konstans hamis értékkal tér vissza, hiszen a mező nem lehet ragadóssé tenni.

### 8.1.3 Game

- **Felelősség**

A statikus Game osztály felelőssége a játék mindenkorai állapotának a tárolása, irányítása. Ebbe bele kell érteni a játékosok listájának jegyzését, a mezők számontartását, az éppen soron lévő játékos és a csapatok pontjainak számontartását. Eltárolja ezen kívül, hogy a jelenlegi aktív játékosnak hány akciót pontja van még hátra a körben. Az osztály továbbá egy felületet biztosít a felhasználóknak, akik (valamelyen köztes felületen keresztül) a játék különböző funkcióit el tudják indítani.

- **Ősosztályok**

-

- **Interfészek**

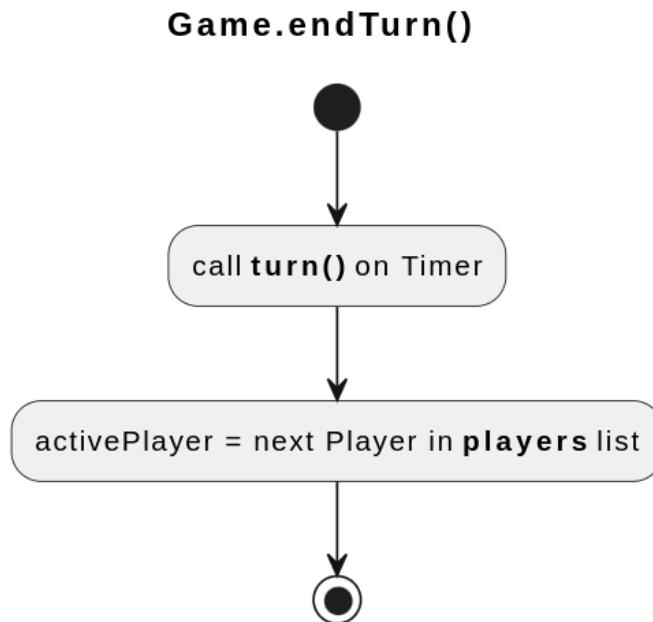
-

- **Attribútumok**

- **-List<Field> fields:** A játék pályaképe, az azon lévő összes mezőt számontartó lista.
- **-List<Player> players:** A játékban résztvevő összes játékos számontartó lista.
- **-Player activePlayer:** A jelenleg aktív játékos referencia. Ez kötelezően a *players* lista egy eleme.
- **-int saboteurPoints:** A szabotőr csapat pontjainak száma. Amennyiben egy kör végén eléri a nyeréshez szükséges számot, a játék megáll.
- **-int mechanicPoints:** A szerelő csapat pontjainak száma, amennyiben egy kör végén eléri a nyeréshez szükséges számot, a játék megáll.
- **-int actionNumber:** Ebben a körben még hány akciót tud végrehajtani a jelenlegi aktív játékos (*activePlayer*). Amennyiben ez a szám 0, az aktív játékos csak akkor fog tudni legközelebb akciót végrehajtani, amikor ismét sorra kerül.

- **Metódusok**

- **+void play():** A játékot indító metódus. Beállítja a jelenlegi aktív játékosat a *players* tömb első elemére.
- **+void end():** Megállítja a játékot (jelen iteráció szerint a metódus nem csinál semmit, azon kívül, hogy leállítja a programot).
- **+void endTurn():** A kör váltást megkezdő metódus. Meghívja a *Timer* osztály *turn()* függvényét, majd a soron következő játékos lesz aktív.



- `+void breakField()`: Meghívja a jelenlegi aktív játékoson (`activePlayer`) a `breakField()` metódust.
- `+void repairField()`: Meghívja a jelenlegi aktív játékoson (`activePlayer`) a `repairField()` metódust.
- `+void place()`: Meghívja a jelenlegi aktív játékoson (`activePlayer`) a `place()` függvényt.
- `+void pickup()`: Meghívja a jelenlegi aktív játékoson (`activePlayer`) a `pickup()` függvényt.
- `+void actionTaken()`: Csökkenti a körben hátralévő akciók számát (`actionNumber`) 1-el, amennyiben az nem 0.
- `+void addMechanicPoints()`: Ad 1 pontot a szerelőknek, azaz növeli a `mechanicPoints` értékét 1-el.
- `+void addSaboteurPoints()`: Ad 1 pontot a szabotőröknek, azaz növeli a `saboteurPoints` értékét 1-el..
- `+void changePumpDirection(Pipe in, Pipe out)`: Meghívja az aktív játékos (`activePlayer`) `changePumpDirection()` függvényét és ennek átadja a kapott paramétereket azonos sorrendben.
- `+void movePipe(Pipe p, Field oldEnd, Field newEnd)`: Átállítja a kiválasztott cső (`p`) egyik végét (`oldEnd`) egy másik helyre (`newEnd`). Fontos, hogy definiált viselkedés, hogy amennyiben a `newEnd` `null`, ez azt jelenti, hogy a hívó nem köti sehol a csövet csak lecsatlakoztatja azt.

#### o pszeudokód

```

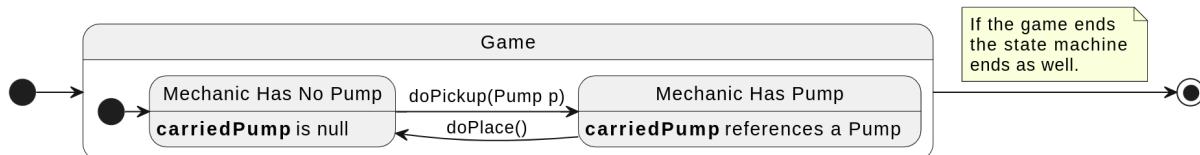
függvény movePipe(p: Pipe, oldEnd: Field, newEnd: Field): üres
    oldEnd->removeNeighbor(p)
    p->removeNeighbor(oldEnd)
    HA newEnd nem null:
        p->addNeighbor(newEnd)
        newEnd->addNeighbor(p)
    Game->actionTaken()
függvény vége
  
```

- **+void makeSlippery():** Meghívja az *activePlayer* tagváltozón a *makeSlippery()* metódust.
- **+void makeSticky():** Meghívja az *activePlayer* tagváltozón a *makeSticky()* metódust.

### 8.1.4 Mechanic

- **Felelősség**

A Player osztályhoz képest plusz felelőssége, hogy számontartja, hogy van-e a karakternél pumpa, és ezt le tudja rakni (feltéve, hogy olyan mezőn áll, ahol ez lehetséges). Lerakásnál felelőssége, hogy az újonnan létrejött komponensek egymással helyesen legyenek összekötve. Pumpát felvenni is tud, ha olyan mezőn áll, ahol ilyet lehet. Emellett javítani is tud mezőket.



- **Ősosztályok**

*Player*

- **Interfészek**

-

- **Attribútumok**

- **-Pump carriedPump:** A szerelőnél lévő pumpa referenciája. Amennyiben ez a változó *null*, az azt jelenti, hogy nincs a szerelőnél pumpa.

- **Metódusok**

- **+void place():** Ha van nála pumpa, azaz a *carriedPump* értéke nem *null*, meghívja az *activeField* tagváltozóján a *placePump()* metódust, paraméterben átadva önmagát. Ez a játék szempontjából azt jelenti, hogy megpróbálja letenni a pumpáját a mezőre, amin éppen áll.
- **+void pickup():** Ha nincs nála pumpa, azaz a *carriedPump* értéke *null*, meghívja az *activeField* tagváltozóján a *pickupPump()* metódust, paraméterben átadva önmagát. Ez a játék szempontjából azt jelenti, hogy ha nincs nála pumpa, megpróbál felvenni egyet a mezőről, amin éppen áll.
- **+void doPickup(Pump p):** A paraméterként kapott pumpát felveszi, azaz beállítja a *carriedPump* változót a paraméterként kapott pumpára, és csökkenti a körben hátralevő akciók számát 1-el, azaz meghívja a *Game* osztály *actionTaken()* metódusát.

#### Pump.doPickup(p)



- **+void doPlace():** Ha eddig volt pumpája, most már nem lesz, lerakja a jelenlegi mezőjére, úgy, hogy amin áll, azt 2 csővel és közöttük 1

*pumpával helyettesíti és csökkenti a körben hátralevő akciók számát  
1-el.*

- o **pszeudokód**

```

függvény doPlace(): üres
    activeField->getNeighbors(): [n1, n2]
    n1->removeNeighbor(activeField)
    n2->removeNeighbor(activeField)
    pipe1: új Pipe
    pipe2: új Pipe
    carriedPump->addNeighbor(pipe1)
    carriedPump->addNeighbor(pipe2)
    carriedPump->changePumpDirection(pipe1, pipe2)
    n1->addNeighbor(pipe1)
    n2->addNeighbor(pipe2)
    pipe1->addNeighbor(carriedPump)
    pipe1->addNeighbor(n1)
    pipe2->addNeighbor(n2)
    pipe2->addNeighbor(carriedPump)
    fields = Game->getFields()
    fields->Remove(activeField)
    fields->Add(pipe1, pipe2, carriedPump)
    periodics = Timer->getPeriodics()
    periodics->Add(carriedPump, pipe1, pipe2)
    periodics->Remove(activeField)
    statefuls = Timer->getStatefuls()
    statefuls->Remove(activeField)
    statefuls->Add(pipe1, pipe2)
    activeField = carriedPump
    carriedPump = null
    Game->actionTaken()
függvény vége

```

- **+void repairField():** Meghívja az *activeField* mezőn a *repair()* függvényt. Ez annak felel meg, hogy megpróbálja megjavítani azt a mezőt, amin éppen áll.

### 8.1.5 Periodic

- **Felelősség**

*Lehetővé teszi hogy a hálózat objektumai sorban meghívodjanak egy-egy körben. Ezalatt, minden egyik típusától függően definiál egy viselkedést (a *step()* függvényben). Ezen viselkedések összessége alkotja a hálózat folyamát.*

- **Ősosztályok**

-

- **Interfészek**

-

- **Attribútumok**

-

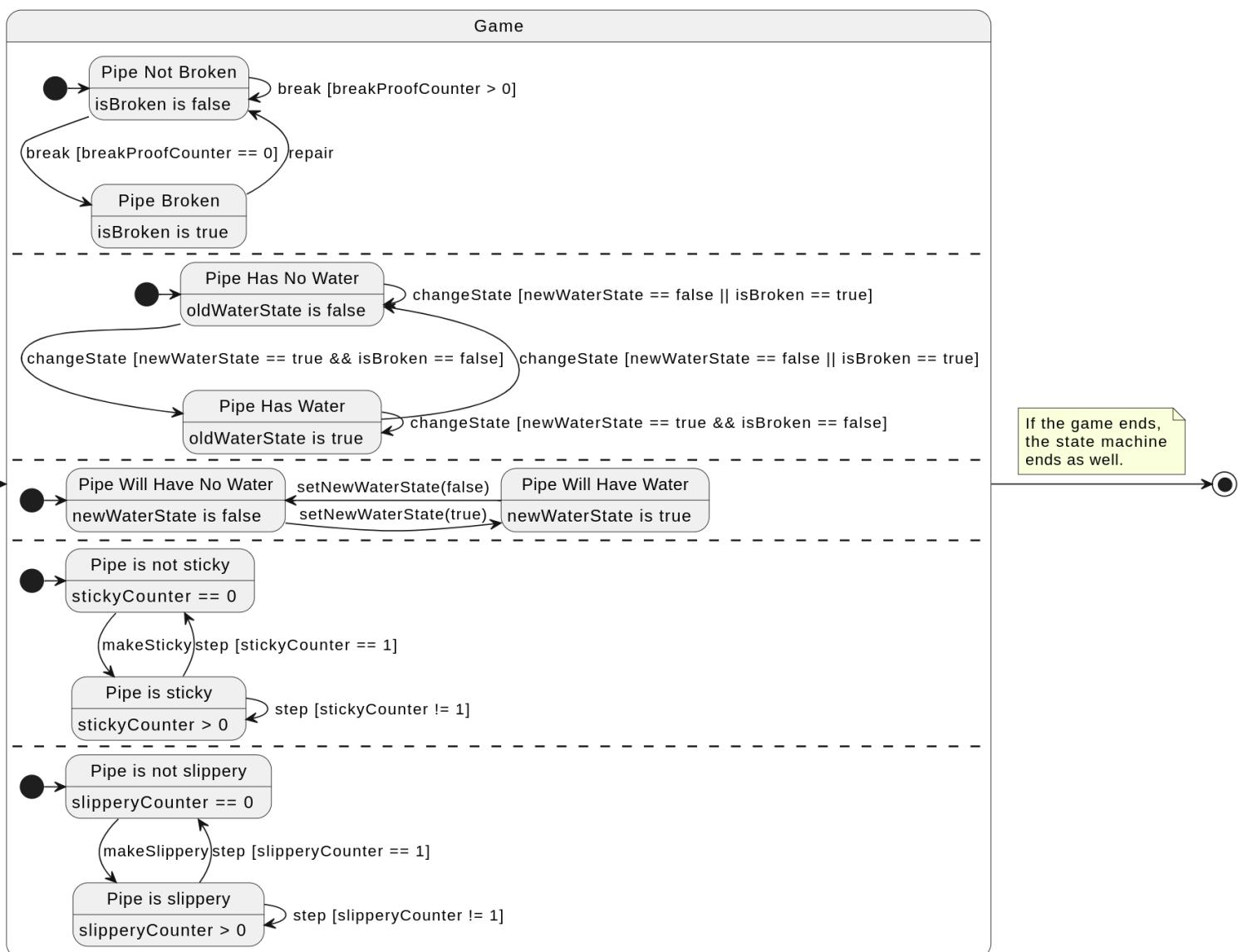
- **Metódusok**

- **+void step()**: minden objektum ami implementálja, ebbe írja le a sajátos viselkedését. Interfész szinten nincs definiálva a metódusnak törzse

### 8.1.6 Pipe

- **Felelősség**

*Felelőssége, hogy számolatrsa, van-e benne víz, el van-e törve, ragadós-e, csúszós-e illetve, hogy el lehet-e törni. Lehetőséget biztosít arra, hogy ezeket a tulajdonságait kívülről lehessen változtatni a játékszabályokban megfogalmazott módon. Az is felelőssége, hogy ezen tulajdonságok függvényében más-más módon regáljon a különböző interakciókra (amikor rá próbálnak lépni, el akarják törni vagy meg akarják javítani).*



- Ősosztályok

*Field*

- Interfészek

*Periodic, Stateful*

- Attribútumok

- **-bool oldWaterState:** azt jegyzi le hogy ha van benne víz a legutolsó kör kezdete óta.
- **-bool newWaterState:** minden körben az **updateState()** hívásra ennek a változónak az értékét átmásolja az **oldWaterState** változóba.
- **-bool isBroken:** **true**, ha el van törve a cső, **false**, ha nincs. Eltörött állapotban a csőnek más a viselkedése, mint ép állapotban.
- **-int breakProofCounter:** Az itt tárolt szám azt jelzi, hogy a jelenlegitől számítva hány kör mülva lehet újra eltörni a csövet.

- Metódusok

- **+void step():** A **breakProofCounter**, a **slipperyCounter** és a **stickyCounter** értékét csökkenti 1-gyel, ha még nem 0 az értékük.
- **+void accept(Player p):** Ha nem áll rajta más játékos (**players** listában nincs objektum) és a paraméterben kapott játékos egy mellette lévő mezőn áll, akkor önmagára lépteti őt és hozzáadja a **players** listához.

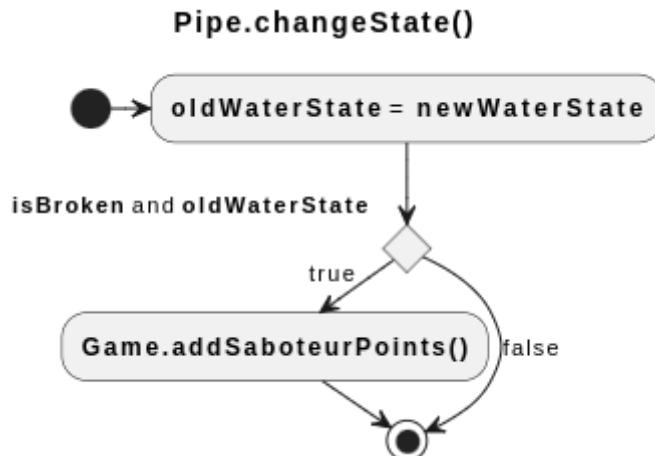
- pszeudokód

```

függvény accept(p: Player): üres
    HA a players lista üres
        p->getActiveField(): oldField
        HA neighbors listában szerepel oldField
            HA slipperyCounter > 0
                n: a neighbors egy random eleme
                p->swapField(n)
                n->addPlayer(p)
            KÜLÖNBEN
                p->swapField(this)
                players->Add(p)
függvény vége

```

- **+void changeState():** Ha különbözik a két állapot (**newWaterState**) és (**oldWaterState**), akkor az **oldWaterState** megkapja a **newWaterState** értékét. Ezután, ha az **oldWaterState** igaz és a cső lyukas, a szabotőrök kapnak pontot.



- **+void repair():** Az **isBroken** értéke hamis lesz és amennyiben igaz volt, csökkenti a hátralévő akciók számát 1-el, úgy, hogy meghívja a **Game** osztály **actionTaken()** metódusát. Ezt követően a **breakProofCounter** értékét egy véletlenszerű pozitív értékre beállítja.

- **pszeudokód**

```

függvény repair(): üres
    HA isBroken == true
        Game->actionTaken()
        breakProofCounter = random érték 1-től 5-ig

    isBroken = false
függvény vége
  
```

- **+void break():** Ha a **breakProofCounter** értéke 0, az **isBroken** értéke igaz lesz és amennyiben hamis volt, csökkenti a hátralévő akciók számát 1-el.

- **pszeudokód**

```

függvény break(): üres
    HA breakProofCounter == 0
        HA isBroken == false
            Game->actionTaken()
            isBroken = true
függvény vége
  
```

- **+void placePump(Mechanic m):** A paraméterül kapott szerelő játékoson meghívja a **doPlace()** metódust (lerakatja vele a pumpát).
- **+void flowWater(Cistern c):** A paraméterül kapott ciszternán meghívja az **addWater()** függvényt, ha az **oldWaterState** igaz ("folyatja bele a vizet").
- **+void giveWater(Pump p):** Meghívja a paraméterként kapott pumpán a **setWater()** függvényt és paraméterként átadja az **oldWaterState** értékét.
- **void makeSlippery():** A **slipperyCounter** értékét 5-re állítja és meghívja a **Game** osztály **actionTaken()** metódusát.
- **void makeSticky():** Ha a **stickyCounter** értéke 0, a **stickyCounter** értékét 5-re állítja és meghívja a **Game** osztály **actionTaken()** metódusát, egyébként nem csinál semmit.
- **bool isSticky():** Ha a **stickyCounter** értéke szigorúan kevesebb, mint 5 és szigorúan nagyobb, mint 0, **true** értékkal tér vissza, egyébként **false**-al.

### 8.1.7 Player

- **Felelősség**

Absztrakt osztály, mely a játékos karakterek ősosztálya, tehát ez az osztály definiálja a szerelő és szabotör osztályok közös funkcióit. Felelőssége, hogy számoltartsa a játékos aktuális tartózkodási mezőjét, és lehetővé tegye a mozgást a hálózaton.

- **Ósosztályok**

- **Interfészek**

- **Attribútumok**

- **#Field activeField:** Az aktuális tartózkodási mező

- **Metódusok**

- **+void breakField():** Meghívja az **activeField** tagváltozón a **break()** metódust
- **+void repairField():** A metódus implementációja ebben az osztályban üres
- **+void pickup():** A metódus implementációja ebben az osztályban üres
- **+void place():** A metódus implementációja ebben az osztályban üres
- **+void swapField(Field f):** Az **activeField** tagváltozóján meghívja a **removePlayer()** metódust **this** paraméterrel, beállítja az **activeField**-et a paraméterként kapott mezőre, végül meghívja a **Game** osztály **actionTaken()** metódusát.

- **pszeudokód**

- 

```
függvény swapField(Field f): üres
    activeField.removePlayer(this)
    activeField = f
    Game.actionTaken()
függvény vége
```

- **+void moveToField(Field f):** Lekérdezi az **activeField** tagváltozótól, hogy ragadós-e, és ha nem, akkor meghívja a paraméterként kapott mezőn az **accept()** metódust **this** paraméterrel.

- **pszeudokód**

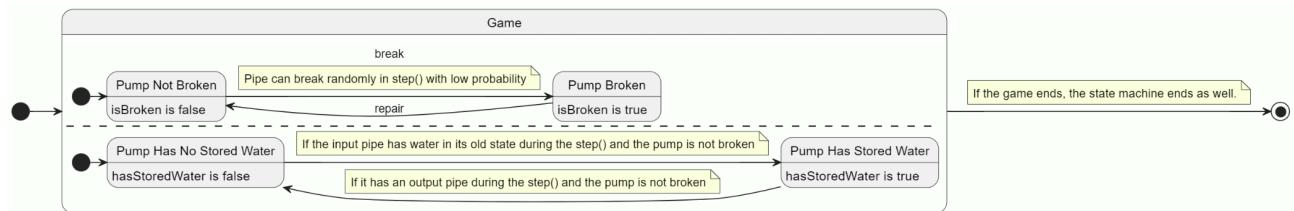
```
függvény moveToField(Field f): üres
    HA activeField.isSticky() == Hamis
        f.accept(this)
függvény vége
```

- **+void changePumpDirection(Pipe in, Pipe out):** Meghívja az **activeField** tagváltozón a megegyező nevű metódust ugyanazzal a paraméterezzel.
- **+void makeSlippery():** A metódus implementációja ebben az osztályban üres.
- **+void makeSticky():** Meghívja az **activeField** tagváltozón a **makeSticky** metódust

## 8.1.8 Pump

- **Felelősség**

Felelőssége, hogy számoltartsa a kimeneti és bemeneti csövét, valamint, hogy van-e a tartályában víz, és hogy törött-e. Emellett a víz folyásáért felelős, tehát a bemeneti csövről a tartályába, a tartályából a kimeneti csövébe pumpálja a vizet, ha nem törött.



- **Ósosztályok**

*Field*

- **Interfészek**

*Periodic*

- **Attribútumok**

- **-Pipe input:** a bemeneti cső referenciája, ennek az állapota alapján dönt, hogy tud-e vizet szívni a tartályába
- **-Pipe output:** a kimeneti cső referenciája, ennek az állapotát állítja be a tartályának állapota alapján lépésre
- **-bool hasStoredWater:** azt jegyzi, hogy tárol-e vizet a tartályában, melyet a kimeneti csövébe tud pumpálni lépésre
- **-bool isBroken:** azt jegyzi, hogy el van-e romolva. Ha el van romolva, akkor nem tud pumpálni lépésre

- **Metódusok**

- **+void step():** Kis valószínűséggel beállítja az **isBroken** tagváltozót True-ra. Ha **isBroken** hamis, akkor az **output** tagváltozón meghívja a **setNewWaterState()** metódust a **hasStoredWater**-rel paraméterként. Ezután az **input** tagváltozón meghívja a **giveWater()** metódust **this**-szel paraméterként.

- **pszeudokód**

```

függvény step(): üres
    isBroken véletlenszerűen beállítódik Igazra
    HA isBroken == Hamis:
        output.setNewWaterState(hasStoredWater)
        input.giveWater(this)
függvény vége
  
```

- **+void changePumpDirection(Pipe in, Pipe out):** Beállítja az **input** és **output** tagváltozókat a paraméterként kapott csövekre, amennyiben ezek nem egyeznek meg, majd meghívja a **Game** osztály **actionTaken()** metódusát.
- **+void repair():** Ha az **isBroken** tagváltozó a metódus meghívása előtt Igaz volt, akkor beállítja Hamisra és meghívja a **Game** osztály **actionTaken()** metódusát.
- **+void setWater(boolean b):** Beállítja a **hasStoredWater** tagváltozó értékét a paraméterként kapott értékre.
- **+void removeNeighbor(Field f):** meghívja a szülő (**Field**) osztály **removeNeighbor(Field f)** metódusát, paraméterként átadva neki f-et, majd ha f megegyezik az **output** tagváltozójával, az **output** értékét null-ra állítja. Ha pedig f megegyezik az **input** tagváltozójával, az **input** értékét állítja null-ra.
  - **pszeudokód**

```

függvény removeNeighbor(Field f): üres
    Field->removeNeighbor(f)
    HA f == output
        output = null
    KÜLÖNBEN HA f == input
        input = null
függvény vége
  
```

## 8.1.9 Saboteur

- **Felelősség**

A Player ōsosztályához képest plusz plusz felelősség, hogy csúszóssá tudja tenni az aktuális mezőjét.

- **Ósosztályok**

Player

- **Interfészek**

-

- **Attribútumok**

-

- **Metódusok**

- **+void makeSlippery():** Meghívja az **activeField** tagváltozón a **makeSlippery()** nevű metódust paraméterek nélkül, majd a **Game** osztály **actionTaken()** metódusát.

## 8.1.10 Source

- **Felelősség**

Felelőssége, hogy számoltartsa a kimenetét (**output: Pipe**) és minden lépésre azon meghívja a **setNewWaterState()** metódust, ezzel vizet pumpálva abba minden körben.

- **Ósosztályok**

Field

- **Interfészek**

Periodic

- **Attribútumok**

- **-Pipe output:** A kimeneti cső referenciája, melyet az addNeighbor() és a removeNeighbor() metódusokkal lehet manipulálni.

- **Metódusok**

- **+void step():** Meghívja az **output** tagváltozón a **setNewWaterState()** metódust True paraméterrel, amennyiben az nem **null**
- **void addNeighbor(Pipe p):** meghívja a szülő (**Field**) osztály **addNeighbor(Field f)** metódusát, paraméterként átadva neki **p-t**, majd az **output** tagváltozóját beállítja **p-re**.
- **void removeNeighbor(Field f):** meghívja a szülő osztály **removeNeighbor(Field f)** metódusát, paraméterként átadva neki **f-et**, majd az **output** tagváltozóját beállítja **null-ra**.

### 8.1.11 Stateful

- **Felelősség**

Lehetővé teszi, hogy a hálózat állapottal rendelkező elemei (**Pipe**) az állapotukat frissítsék. Azért szükséges, hogy a hálózat elemeinek léptetése és az állapotuk frissítése különválasztható legyen, előbb fusson le az összes elem léptetése, mint hogy az állapotokat frissitenénk. Az előbbiért a **Periodic** interface **step()** metódusa felelős, az utóbbiért a **Stateful** interface **changeState()** metódusa.

- **Ősosztályok**

-

- **Interfészek**

-

- **Attribútumok**

-

- **Metódusok**

- **+void changeState():** minden implementáló osztály ebben definiálja az állapot frissítéséhez szükséges lépésekét

### 8.1.12 Timer

- **Felelősség**

Felelőssége, hogy számoltartsa a **Periodic** és **Stateful** interface-eket megvalósító objektumokat, és ezekre minden **turn()** hívásra meghívja először az összes **Periodic** objektum **step()** metódusát, majd az összes **Stateful** objektum **changeState()** metódusát.

- **Ősosztályok**

-

- **Interfészek**

-

- **Attribútumok**

- **-List<Periodic> periodics:** Az összes **Periodic** interface-t megvalósító objektumot tartalmazó lista
- **-List<Stateful> statefuls:** Az összes **Stateful** interface-t megvalósító objektumot tartalmazó lista

- **Metódusok**

- **+void turn():** Meghívja a **periodics** listában szereplő összes objektumra a **step()** metódust, majd a **statefuls** lista összes elemére meghívja a **changeState()** metódust
  - **pszeudokód**

```

függvény turn(): üres
    MINDEN periodics-beli p-re:
        p.step()
    MINDEN statefuls-beli s-re:
        s.changeState()
függvény vége
  
```

## 8.2 A tesztek részletes tervei, leírásuk a teszt nyelvén

A tesztek elején többször szerepel a *load* parancs. Ez ki fogja írni a beolvásott (kiindulási) pályát a kimeneti nyelven, ezt mi minden tesztesetnél a parancs kiadása után beírtuk, ez nem azt jelenti, hogy ezt be kell írni, csak így jobban látszik, hogy mi változik.

### 8.2.1 Az aktív játékos sikeresen mozog

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvásunk egy mozgató parancsot, a játékos rálép a célként megadott mezőre és veszít egy akciót pontot. A mozgás azért sikeres, mert az adott mezőre lehet lépni (a célpont szomszédos és cső esetén nem áll rajta más) és a játékosnak van még akciót pontja ebben a körben illetve el tud lépni a mezőjéről (nincs beleragadva).

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív játékos mozgatása, illetve az ehhez szükséges követelmények ellenőrzése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a játékos kiinduló mezőn marad
4. nem csökken az actionNumber

- **Bemenet**

```

output
load -f tests/map/map1.txt
(
"Game#0":
{
  "players": [
    "0:Mechanic#1":
```

```

{
    "activeField:Pipe#2":
    {
        "neighbors":
        [
            "0:Cistern#3":
            {
                "neighbors":
                [
                    "0@[Reference)": Pipe#2
                ],
                "players":
                [],
                "input@[Reference)": Pipe#2
            }
        ],
        "players":
        [
            "0@[Reference)": Mechanic#1
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
},
],
"fields":
[
    "0@[Reference)": Pipe#2,
    "1@[Reference)": Cistern#3
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#4":
{
    "periodics":
    [
        "0@[Reference)": Cistern#3,
        "1@[Reference)": Pipe#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}
)
y

```

- Elvárt kimenet

"Game#0":

```
{
  "players": [
    {
      "0:Mechanic#1": {
        "activeField:Cistern#2": {
          "neighbors": [
            {
              "0:Pipe#3": {
                "neighbors": [
                  "0@[Reference)": Cistern#2
                ],
                "players": [],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
              }
            ],
            "players": [
              "0@[Reference)": Mechanic#1
            ],
            "input@[Reference)": Pipe#3
          }
        }
      }
    ],
    "fields": [
      {
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Cistern#2
      },
      "saboteurPoints": 0,
      "mechanicPoints": 0,
      "actionNumber": 2,
      "activePlayer@[Reference)": Mechanic#1
    },
    "Timer#4": {
      "periodics": [
        {
          "0@[Reference)": Cistern#2,
          "1@[Reference)": Pipe#3
        },
        "statefuls": [
          {
            "0@[Reference)": Pipe#3
          }
        ]
      }
    }
  ]
}
```

## 8.2.2 Az aktív játékos csúszós csőre lép

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy mozgató parancsot, a játékos rálép a célként megadott csőre és veszít egy akciót pontot. Mivel a célként választott cső csúszós, a játékos a cső egy véletlenszerűen választott szomszédjára kerül. A mozgás sikeres, mert az adott csövön nem állt senki, a szomszédai pedig nem lehetnek csövek, tehát nem fordulhat elő, hogy nem tud rájuk lépni a játékos.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos mozgatása, az ehhez szükséges követelmények ellenőrzése, csúszós cső viselkedése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a játékos a csúszós csövön áll, vagy rossz oldalra csuszik
4. nem csökken az actionNumber

- Bemenet

output

random

//legyen kikapcsolva a random tényező

load -f tests/map/map2.txt

(

"Game#0":

{

    "players":

[

        "0:Mechanic#1":

{

            "activeField:Cistern#2":

{

                "neighbors":

[

                    "0:Pipe#3":

{

                        "neighbors":

[

                            "0@[Reference)": Cistern#2,

                            "1:Pump#4":

{

                                "neighbors":

[

                                    "0@[Reference)": Pipe#3

],

                        "players":

[]

                        "output@[Reference)": Pipe#3,

                        "hasStoredWater": false,

                        "isBroken": false

}

```

        ],
        "players": [
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 4,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
],
"players": [
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3
}
},
"fields": [
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#5":
{
    "periodics": [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls": [
        "0@[Reference)": Pipe#3
    ]
}
)
play
move -f Pipe#3
//a kikapcsolt random miatt a program megkérdezi, hogy melyik oldalra csússzon a karakter,
a válasz legyen "regi", de "uj" - al is működik


- Elvárt kimenet


"Game#0":
```

```
{
  "players": [
    {
      "0:Mechanic#1": {
        "activeField:Cistern#2": {
          "neighbors": [
            {
              "0:Pipe#3": {
                "neighbors": [
                  {
                    "0@[Reference)": Cistern#2,
                    "1:Pump#4": {
                      "neighbors": [
                        {
                          "0@[Reference)": Pipe#3
                        },
                        "players": [],
                        "output@[Reference)": Pipe#3,
                        "hasStoredWater": false,
                        "isBroken": false
                      }
                    ],
                    "players": []
                  }
                ]
              }
            },
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 4,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
          }
        ],
        "players": [
          {
            "0@[Reference)": Mechanic#1
          }
        ],
        "input@[Reference)": Pipe#3
      }
    }
  ],
  "fields": [
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
  ]
}
```

```

        ],
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 2,
        "activePlayer@[Reference)": Mechanic#1
    },
    "Timer#5":
    {
        "periodics":
        [
            "0@[Reference)": Cistern#2,
            "1@[Reference)": Pump#4,
            "2@[Reference)": Pipe#3
        ],
        "statefuls":
        [
            "0@[Reference)": Pipe#3
        ]
    }
}

```

### 8.2.3 Az aktív játékos nem tud lépni a kívánt mezőre

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy mozgató parancsot, a játékos megpróbál a célként megadott mezőre lépni, de nem tud a következő okok valamelyike miatt: a célként megadott mező egy cső, amin már állnak, a játékos ragadós csövön áll, a játékosnak nincs több akciót pontja, vagy a célként megadott mező nem szomszédos az aktuális tartózkodási hellyel.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív játékos mozgatása, ragadós cső viselkedése, foglalt cső viselkedése, akciót pontok kezelése, szomszédosság ellenőrzése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a játékos rá tud lépni a mezőre amikor nem lenne szabad
4. csökken az actionNumber, pedig nem kéne

- **Bemenet**

```

output
load -f tests/map/map3.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":

```

```

        {
            "neighbors":
            [
                "0@[Reference)": Cistern#2
            ],
            "players":
            [
                "0:Saboteur#4":
                {
                    "activeField@[Reference]":
                    Pipe#3
                }
            ],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
        }
    ],
    "players":
    [
        "0@[Reference)": Mechanic#1
    ],
    "input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#4
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#5":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}

```

```

        ]
    }
)
play
move -f Pipe#3
● Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Cistern#2
                        ],
                        "players":
                        [
                            "0:Saboteur#4":
                            {
                                "activeField@[Reference]":
                                "0@[Reference]": Mechanic#1
                            },
                            "input@[Reference)": Pipe#3
                        ]
                    },
                    "1@[Reference)": Saboteur#4
                ],
                "fields":
                [
                    "0@[Reference)": Pipe#3,

```

```

        "1@[Reference)": Cistern#2
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Mechanic#1
},
"Timer#5":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}
```

## 8.2.4 A játékos sikeresen kilyukaszt egy csövet

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet kilyukasztó parancsot, melynek következtében a cső, amin a játékos éppen áll, kilyukad. A lyukasztás azért sikeres, mert a cső eddig nem volt eltörve, valamint éppen törhető állapotban van (mert nem volt foltozva, vagy ha volt, akkor már olyan régen, hogy újra lyukasztható lett).

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos csövet lyukaszt.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a cső nem lyukad ki - isBroken: false
3. nem vonódik le akciópont - actionNumber: 3

- Bemenet

```

output
load -f tests/map/map4.txt
(
"Game#0":
{
    "players":
    [
        "0:Saboteur#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [],
                "players":
                [
                    "0@[Reference)": Saboteur#1
                ],

```

```

        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
},
],
"fields":
[
    "0@[Reference)": Pipe#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#1
},
"Timer#3":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}
)
play
breakField
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Saboteur#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [],
                "players":
                [
                    "0@[Reference)": Saboteur#1
                ],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,

```

```

        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": true
    }
},
],
"fields":
[
    "0@[Reference)": Pipe#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 2,
"activePlayer@[Reference)": Saboteur#1
},
"Timer#3":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}
}

```

## 8.2.5 A játékos nem tud kilyukasztani egy csövet

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet kilyukasztó parancsot, de a cső, amelyen a játékos éppen áll, mégsem lyukad ki a következő okok valamelyike miatt: a cső eleve lyukas volt, vagy a cső nemrég volt foltozva és még nem lyukasztható ki újra.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív játékos csövet lyukaszt, lyukasság ellenőrzése, javítási utáni lyukasztás gátolásának ellenőrzése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a cső kilyukad - isBroken: true
3. levonódik egy akciót, pedig nem kéne - actionNumber: 2

- **Bemenet**

```

output
load -f tests/map/map5.txt
(
"Game#0":
{
    "players":
    [
        "0:Saboteur#1":
        {

```

```

"activeField:Pipe#2":
{
    "neighbors": [],
    "players": [
        {
            "0@[Reference)": Saboteur#1
        },
        "oldWaterState": false,
        "breakProofCounter": 1,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
},
],
"fields": [
    {
        "0@[Reference)": Pipe#2
    },
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Saboteur#1
},
"Timer#3":
{
    "periodics": [
        {
            "0@[Reference)": Pipe#2
        },
        "statefuls": [
            {
                "0@[Reference)": Pipe#2
            }
        ]
    }
}
)
play
breakField
    • Elvárt kimenet
"Game#0":
{
    "players": [
        "0:Saboteur#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors": [
                    ...
                ]
            }
        }
    ]
}

```

```

        ],
        "players": [
            [
                "0@[Reference)": Saboteur#1
            ],
            "oldWaterState": false,
            "breakProofCounter": 1,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
        }
    ],
    "fields": [
        [
            "0@[Reference)": Pipe#2
        ],
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 3,
        "activePlayer@[Reference)": Saboteur#1
    ],
    "Timer#3": {
        "periodics": [
            [
                "0@[Reference)": Pipe#2
            ],
            "statefuls": [
                [
                    "0@[Reference)": Pipe#2
                ]
            ]
        }
    }
}

```

## 8.2.6 Szerelő sikeresen megjavít egy mezőt

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy javító parancsot, melynek következtében a mező, amin a szerelő áll, megjavul és a játékos veszít egy akciót pontot. A javítás azért volt sikeres, mert a mező, amin a játékos áll, cső vagy pumpa és valóban el volt romolva.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív szerelő játékos megjavít egy mezőt.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a cső továbbra is el van törve - isBroken: true
3. nem vonódik le akciót pont - actionNumber: 3
4. a breakProofCounter nem állítódik be - breakProofCounter: 0

- **Bemenet**

output

```

random
//kikapcsoljuk
load -f tests/map/map6.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [],
                "players":
                [
                    "0@[Reference)": Mechanic#1
                ],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": true
            }
        }
    ],
    "fields":
    [
        "0@[Reference)": Pipe#2
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Mechanic#1
},
"Timer#3":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}
)
play
repairField

```

```
//azt mondjuk, hogy a breakProofCounter legyen 3
    • Elvárt kimenet
"Game#0":
{
    "players": [
        "0:Mechanic#1": {
            "activeField:Pipe#2": {
                "neighbors": [],
                "players": [
                    {
                        "0@[Reference)": Mechanic#1
                    },
                    "oldWaterState": false,
                    "breakProofCounter": 3,
                    "slipperyCounter": 0,
                    "stickyCounter": 0,
                    "newWaterState": false,
                    "isBroken": false
                }
            }
        ],
        "fields": [
            "0@[Reference)": Pipe#2
        ],
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 2,
        "activePlayer@[Reference)": Mechanic#1
    },
    "Timer#3": {
        "periodics": [
            "0@[Reference)": Pipe#2
        ],
        "statefuls": [
            "0@[Reference)": Pipe#2
        ]
    }
}
```

### 8.2.7 Szerelő nem tud megjavítani egy mezőt

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy javító parancsot, de a mező, amin a játékos áll, nem javul meg a következő okok valamelyike miatt: a mező nem

cső vagy pumpa (tehát eleve nem lehetett elromolva), vagy a mező ugyan cső vagy pumpa, de nem volt elromolva.

- **Ellenőrzött funkciionalitás, várható hibahelyek**

Az aktív szerelő játékos megjavít egy mezőt, pumpa működőképességének ellenőrzése, cső lyukasságának ellenőrzése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a breakProofCounter elállítódik valamelyen nem nulla értékre
3. levonódik action - actionNumber: 2

- **Bemenet**

```
output
load -f tests/map/map7.txt
(
"Game#0":
{
  "players": [
    {
      "0:Mechanic#1": {
        "activeField:Pipe#2": {
          "neighbors": [],
          "players": [
            {
              "0@[Reference)": Mechanic#1
            },
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
          }
        }
      },
      "fields": [
        {
          "0@[Reference)": Pipe#2
        },
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 3,
        "activePlayer@[Reference)": Mechanic#1
      ],
      "Timer#3": {
        "periodics": [
          "0@[Reference)": Pipe#2
        ]
      }
    }
  ]
}
```

```

        ],
        "statefuls":
        [
            "0@[Reference)": Pipe#2
        ]
    }
)
play
repairField
● Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [],
                "players":
                [
                    "0@[Reference)": Mechanic#1
                ],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
            }
        }
    ],
    "fields":
    [
        "0@[Reference)": Pipe#2
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Mechanic#1
},
"Timer#3":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2
    ],
    "statefuls":
    [

```

```

        "0@[Reference)": Pipe#2
    ]
}

```

## 8.2.8 Szabotőr csúszóssá tesz egy csövet

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csúszóssá tevő parancsot, melynek következtében a cső, amelyen a szabotőr áll, a következő öt körben csúszós lesz. A művelet mindenképpen sikeres (feltéve, hogy a játékosnak van akciót pontja), hiszen nem fordulhat elő, hogy a cső már csúszós volt, mert akkor nem állhatna rajta játékos.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív szabotőr játékos csúszóssá tesz egy csövet.

Várható hibahelyek:

- load-ban megadott fájl nem létezik
- a cső nem lesz csúszós - slipperyCounter: 0
- nem vonódik le action - actionNumber: 3

- Bemenet

```

output
load -f tests/map/map8.txt
(
"Game#0":
{
  "players":
  [
    "0:Saboteur#1":
    {
      "activeField:Pipe#2":
      {
        "neighbors":
        [],
        "players":
        [
          "0@[Reference)": Saboteur#1
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
      }
    }
  ],
  "fields":
  [
    "0@[Reference)": Pipe#2
  ],
  "saboteurPoints": 0,
  "mechanicPoints": 0,
  "actionNumber": 3,
}

```

```

    "activePlayer@[Reference)": Saboteur#1
},
"Timer#3":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}
)
play
makeSlippery
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Saboteur#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [],
                "players":
                [
                    "0@[Reference)": Saboteur#1
                ],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 5,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
            }
        }
    ],
    "fields":
    [
        "0@[Reference)": Pipe#2
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 2,
    "activePlayer@[Reference)": Saboteur#1
},
"Timer#3";

```

```
{
    "periodics": [
        "0@[Reference)": Pipe#2
    ],
    "statefuls": [
        "0@[Reference)": Pipe#2
    ]
}
```

### 8.2.9 Játékos ragadóssá tesz egy csövet

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy ragadóssá tevő parancsot, melynek következtében a cső, amelyen a játékos áll, a következő öt körben ragadós lesz.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos ragadóssá tesz egy csövet.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a játékos nem tudja ragadóssá tenni a csövet amin áll, pedig kéne tudnia

- Bemenet

```
output
load -f tests/map/map9.txt
(
"Game#0":
{
    "players": [
        "0:Saboteur#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0:Cistern#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                                "players":
                                [],
                                "input@[Reference)": Pipe#3
                            },
                            "1@[Reference)": Pump#2
                        ],

```

```

        "players":  

        []],  

        "oldWaterState": false,  

        "breakProofCounter": 0,  

        "slipperyCounter": 0,  

        "stickyCounter": 0,  

        "newWaterState": false,  

        "isBroken": false  

    }]  

    ],  

    "players":  

    [  

        "0@[Reference)": Saboteur#1  

    ],  

    "output@[Reference)": Pipe#3,  

    "hasStoredWater": false,  

    "isBroken": false  

    }]  

],  

"fields":  

[  

    "0@[Reference)": Pipe#3,  

    "1@[Reference)": Cistern#4,  

    "2@[Reference)": Pump#2  

],  

"saboteurPoints": 0,  

"mechanicPoints": 0,  

"actionNumber": 3,  

"activePlayer@[Reference)": Saboteur#1  

},  

"Timer#5":  

{  

    "periodics":  

[  

    "0@[Reference)": Cistern#4,  

    "1@[Reference)": Pump#2,  

    "2@[Reference)": Pipe#3  

],  

"statefuls":  

[  

    "0@[Reference)": Pipe#3  

]  

}  

play  

makeSticky  


- Elvárt kimenet


"Game#0":  

{
    "players":  

[  


```

```

    "0:Saboteur#1":
    {
        "activeField:Pipe#2":
        {
            "neighbors":
            [
                "0:Cistern#3":
                {
                    "neighbors":
                    [
                        "0@[Reference)": Pipe#2
                    ],
                    "players":
                    [],
                    "input@[Reference)": Pipe#2
                },
                "1:Pump#4":
                {
                    "neighbors":
                    [
                        "0@[Reference)": Pipe#2
                    ],
                    "players":
                    [
                        "0@[Reference)": Saboteur#1
                    ],
                    "output@[Reference)": Pipe#2,
                    "hasStoredWater": false,
                    "isBroken": false
                }
            ],
            "players":
            [],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 5,
            "newWaterState": false,
            "isBroken": false
        }
    },
    "fields":
    [
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Cistern#3,
        "2@[Reference)": Pump#4
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 2,
    "activePlayer@[Reference)": Saboteur#1
},
"Timer#5":
{

```

```

"periodics":
[
    "0@[Reference)": Cistern#3,
    "1@[Reference)": Pump#4,
    "2@[Reference)": Pipe#2
],
"statefuls":
[
    "0@[Reference)": Pipe#2
]
}

```

## 8.2.10 Játékos befejezi a körét

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy kör vége parancsot, melynek következtében az aktív játékos a következő játékos lesz, valamint a játék átáll az egy körrel későbbi állapotba (tehát végbemennek a változások, amik a kört befejező játékos körében történtek). A konkrét végbement változások: a stickyCounter illetve a slipperyCounter attribútumok értéke eggyel csökkent (ha 0-nál nagyobb volt), a vízfolyás haladt egy körrel, ennek következtében a csapatok pontokat kaphattak.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos befejezi a körét.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. nem változik meg az activePlayer

- Bemenet

```

output
load -f tests/map/map10.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Cistern#2,
                            "1:Pump#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                                "players":
                                [
                                    "0:Saboteur#5":

```

```

        }

    "activeField@[Reference)": Pump#4
    }
    ],
    "output@[Reference)": Pipe#3,
    "hasStoredWater": true,
    "isBroken": false
}
],
"players":
[],
"oldWaterState": true,
"breakProofCounter": 0,
"slipperyCounter": 4,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
)
play

```

```

endTurn
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Cistern#2,
                            "1:Pump#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                                "players":
                                [
                                    "0:Saboteur#5":
                                    {
                                        "activeField@[Reference)": Pump#4
                                        }
                                    ],
                                    "output@[Reference)": Pipe#3,
                                    "hasStoredWater": true,
                                    "isBroken": false
                                }
                            ],
                            "players":
                            [],
                            "oldWaterState": true,
                            "breakProofCounter": 0,
                            "slipperyCounter": 3,
                            "stickyCounter": 0,
                            "newWaterState": true,
                            "isBroken": false
                        }
                    ],
                    "players":
                    [
                        "0@[Reference)": Mechanic#1
                    ],

```

```

        "input@[Reference)": Pipe#3
    }
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 1,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}
```

## 8.2.11 Sticky- és slipperyCounter csökken a kör végén

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy kör vége parancsot, melynek következtében az aktív játékos a következő játékos lesz, valamint a játék átáll az egy körrrel későbbi állapotba (tehát végbemennek a változások, amik a kört befejező játékos körében történtek). A konkrét végbement változások: a stickyCounter illetve a slipperyCounter attribútumok értéke eggyel csökkent (ha 0-nál nagyobb volt), a vízfolyás haladt egy körrrel, ennek következtében a csapatok pontokat kaphattak.

- Ellenőrzött funkcionalitás, várható hibahelyek

A stickyCounter csökkenése, slipperyCounter csökkenése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. stickyCounter és/vagy slipperyCounter nem csökken pedig értéke nagyobb mint nulla

- Bemenet

```

output
load -f tests/map/map11.txt
(
"Game#0":
{
    "players":
```

```
[
    "0:Mechanic#1":
    {
        "activeField:Cistern#2":
        {
            "neighbors":
            [
                "0:Pipe#3":
                {
                    "neighbors":
                    [
                        "0@[Reference)": Cistern#2,
                        "1:Pump#4":
                        {
                            "neighbors":
                            [
                                "0@[Reference)": Pipe#3
                            ],
                            "players":
                            [
                                "0:Saboteur#5":
                                {
                                    "activeField@[Reference)": Pump#4
                                    }
                                ],
                                "output@[Reference)": Pipe#3,
                                "hasStoredWater": true,
                                "isBroken": false
                            }
                        ],
                        "players":
                        [],
                        "oldWaterState": true,
                        "breakProofCounter": 0,
                        "slipperyCounter": 4,
                        "stickyCounter": 0,
                        "newWaterState": false,
                        "isBroken": false
                    }
                ],
                "players":
                [
                    "0@[Reference)": Mechanic#1
                ],
                "input@[Reference)": Pipe#3
            }
        },
        "1@[Reference)": Saboteur#5
    ],
    "fields":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Cistern#2,
        "2@[Reference)": Pump#4
    ]
]
```

```

        ],
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 3,
        "activePlayer@[Reference)": Mechanic#1
    },
    "Timer#6":
    {
        "periodics":
        [
            "0@[Reference)": Cistern#2,
            "1@[Reference)": Pump#4,
            "2@[Reference)": Pipe#3
        ],
        "statefuls":
        [
            "0@[Reference)": Pipe#3
        ]
    }
)
play
endTurn
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Cistern#2,
                            "1:Pump#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                                "players":
                                [
                                    "0:Saboteur#5":
                                    {
                                        "activeField@[Reference)": Pump#4
                                    }
                                ],
                                "saboteurPoints": 0,
                                "mechanicPoints": 0,
                                "actionNumber": 3,
                                "activePlayer@[Reference)": Saboteur#5
                            }
                        ],
                        "saboteurPoints": 0,
                        "mechanicPoints": 0,
                        "actionNumber": 3,
                        "activePlayer@[Reference)": Mechanic#1
                    }
                ],
                "saboteurPoints": 0,
                "mechanicPoints": 0,
                "actionNumber": 3,
                "activePlayer@[Reference)": Mechanic#1
            }
        }
    ]
}

```

```

        "output@[Reference)": Pipe#3,
        "hasStoredWater": true,
        "isBroken": false
    }
],
"players":
[],
"oldWaterState": true,
"breakProofCounter": 0,
"slipperyCounter": 3,
"stickyCounter": 0,
"newWaterState": true,
"isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 1,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}

```

### 8.2.12 Csapatok pontot kapnak a kör végén

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy kör vége parancsot, melynek következtében az aktív játékos a következő játékos lesz, valamint a játék átáll az egy körrrel későbbi állapotba (tehát végbemennek a változások, amik a kört befejező játékos körében történtek). A konkrét végbement változások: a stickyCounter illetve a slipperyCounter attribútumok értéke eggyel csökkent (ha 0-nál nagyobb volt), a vízfolyás haladt egy körrrel, ennek következtében a csapatok pontokat kaphattak.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az szerelők pontozása, szabotőrök pontozása.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. szerelő vagy szabotór csapat nem kap pontot, pedig víz jutott a ciszternákba/folyt el a lyukakon

- **Bemenet**

output

load -f tests/map/map12.txt

```
(
"Game#0":
{
  "players": [
    "0:Mechanic#1": {
      "activeField:Cistern#2": {
        "neighbors": [
          "0:Pipe#3": {
            "neighbors": [
              "0@[Reference)": Cistern#2,
              "1:Pump#4": {
                "neighbors": [
                  "0@[Reference)": Pipe#3
                ],
                "players": [
                  "0:Saboteur#5": {
                    }
                  ]
                ],
                "output@[Reference)": Pipe#3,
                "hasStoredWater": true,
                "isBroken": false
              }
            ],
            "players": []
          }
        ]
      }
    }
  ]
}
```

```

        "oldWaterState": true,
        "breakProofCounter": 0,
        "slipperyCounter": 4,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
)
play
endTurn
● Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":

```

```
[
    "0:Pipe#3":
    {
        "neighbors":
        [
            "0@[Reference)": Cistern#2,
            "1:Pump#4":
            {
                "neighbors":
                [
                    "0@[Reference)": Pipe#3
                ],
                "players":
                [
                    "0:Saboteur#5":
                    {
                        "activeField@[Reference)": Pump#4
                        }
                    ],
                    "output@[Reference)": Pipe#3,
                    "hasStoredWater": true,
                    "isBroken": false
                }
            ],
            "players":
            [],
            "oldWaterState": true,
            "breakProofCounter": 0,
            "slipperyCounter": 3,
            "stickyCounter": 0,
            "newWaterState": true,
            "isBroken": false
        }
    ],
    "players":
    [
        "0@[Reference)": Mechanic#1
    ],
    "input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
]
```

```

    "saboteurPoints": 0,
    "mechanicPoints": 1,
    "actionNumber": 3,
    "activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}

```

### 8.2.13 Víz áramlik a kör végén

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy kör vége parancsot, melynek következtében az aktív játékos a következő játékos lesz, valamint a játék átáll az egy körrrel későbbi állapotba (tehát végbemennek a változások, amik a kört befejező játékos körében történtek). A konkrét végbement változások: a stickyCounter illetve a slipperyCounter attribútumok értéke eggyel csökkent (ha 0-nál nagyobb volt), a vízfolyás haladt egy körrrel, ennek következtében a csapatok pontokat kaphattak.

- Ellenőrzött funkcionalitás, várható hibahelyek

A víz áramlása.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. víz nem áramlott

- Bemenet

```

output
load -f tests/map/map13.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Cistern#2,

```

```

    "1:Pump#4":
    {
        "neighbors":
        [
            "0@[Reference)": Pipe#3
        ],
        "players":
        [
            "0:Saboteur#5":
            {
                ...
            }
        ],
        "output@[Reference)": Pipe#3,
        "hasStoredWater": true,
        "isBroken": false
    }
],
"players":
[],
"oldWaterState": true,
"breakProofCounter": 0,
"slipperyCounter": 4,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
    ]
}

```

```

        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
)
play
endTurn
● Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Cistern#2,
                            "1:Pump#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                                "players":
                                [
                                    "0:Saboteur#5":
                                    {
                                        "oldWaterState": true,
                                        "breakProofCounter": 0,
                                        "slipperyCounter": 3,
                                        "stickyCounter": 0,
                                        "hasStoredWater": true,
                                        "isBroken": false
                                    }
                                ],
                                "output@[Reference)": Pipe#3,
                                "hasStoredWater": true,
                                "isBroken": false
                            }
                        ],
                        "players":
                        [],
                        "oldWaterState": true,
                        "breakProofCounter": 0,
                        "slipperyCounter": 3,
                        "stickyCounter": 0,
                        "hasStoredWater": true,
                        "isBroken": false
                    }
                ],
                "output@[Reference)": Pipe#3,
                "hasStoredWater": true,
                "isBroken": false
            }
        }
    ]
}

```

```

        "newWaterState": true,
        "isBroken": false
    }
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Cistern#2,
    "2@[Reference)": Pump#4
],
"saboteurPoints": 0,
"mechanicPoints": 1,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}

```

## 8.2.14 Szerelő sikeresen letesz egy új pumpát

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát letévő parancsot, melynek következetében a cső, amin a szerelő áll, megfeleződik és a két fél közé beékelődik az új pumpa. A művelet azért sikeres, mert a szerelő valóban csövön áll, van nála pumpa és van még akciót pontja.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív szerelő játékos pumpát tesz le.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a szerelő nem tud pumpát letenni, pedig minden feltétel adott a sikerhez
3. rosszak az input-output beállítások

## 4. rosszak a szomszédságok

## ● Bemenet

```

output
load -f tests/map/map14.txt
(
"Game#0":
{
  "players":
  [
    "0:Mechanic#1":
    {
      "activeField:Pipe#2":
      {
        "neighbors":
        [
          "0:Cistern#3":
          {
            "neighbors":
            [
              "0@[Reference)": Pipe#2
            ],
            "players":
            [],
            "input@[Reference)": Pipe#2
          },
          "1:Pump#4":
          {
            "neighbors":
            [
              "0@[Reference)": Pipe#2
            ],
            "players":
            [],
            "output@[Reference)": Pipe#2,
            "hasStoredWater": false,
            "isBroken": false
          }
        ],
        "players":
        [
          "0@[Reference)": Mechanic#1
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
      },
      "carriedPump:Pump#5":
    }
  ]
}

```

```

        {
            "neighbors": [],
            "players": [],
            "hasStoredWater": false,
            "isBroken": false
        }
    },
    "fields": [
        {
            "0@[Reference)": Cistern#3,
            "1@[Reference)": Pump#4,
            "2@[Reference)": Pipe#2
        },
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 3,
        "activePlayer@[Reference)": Mechanic#1
    },
    "Timer#6":
    {
        "periodics":
        [
            {
                "0@[Reference)": Pipe#2,
                "1@[Reference)": Cistern#3,
                "2@[Reference)": Pump#4
            },
            "statefuls":
            [
                {
                    "0@[Reference)": Pipe#2
                }
            ]
        }
    }
)
play
place


- Elvárt kimenet


"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":

```

```
[
    "0@[Reference)": Pump#2,
    "1:Cistern#4":
    {
        "neighbors":
        [
            "0@[Reference)": Pipe#3
        ],
        "players":
        []
    }
],
"players":
[],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
},
"1:Pipe#5":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2,
        "1:Pump#6":
        {
            "neighbors":
            [
                "0@[Reference)": Pipe#5
            ],
            "players":
            [],
            "hasStoredWater": false,
            "isBroken": false
        }
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3,
"output@[Reference)": Pipe#5,
"hasStoredWater": false,
```

```

        "isBroken": false
    }
],
"fields":
[
    "0@[Reference)": Cistern#4,
    "1@[Reference)": Pump#6,
    "2@[Reference)": Pump#2,
    "3@[Reference)": Pipe#3,
    "4@[Reference)": Pipe#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 1,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#7":
{
    "periodics":
    [
        "0@[Reference)": Cistern#4,
        "1@[Reference)": Pump#6,
        "2@[Reference)": Pump#2,
        "3@[Reference)": Pipe#3,
        "4@[Reference)": Pipe#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#5
    ]
}
}

```

## 8.2.15 Szerelő nem tud pumpát letenni

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát letévő parancsot, de nem sikerül a művelet a következő oload kok valamelyike miatt: a szerelő nem csövön áll, a szerelőnél nincs pumpa, a játékosnak nincs több akciót pontja.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív szerelő játékos pumpát tesz le, játékosnál van-e pumpa ellenőrzés.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a szerelő tud pumpát letenni, pedig nem minden feltétel adott a sikereshez

- **Bemenet**

```

output
load -f tests/map/map15.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":

```

```
{
  "activeField:Pipe#2":
  {
    "neighbors":
    [
      "0:Cistern#3":
      {
        "neighbors":
        [
          "0@[Reference)": Pipe#2
        ],
        "players":
        [],
        "input@[Reference)": Pipe#2
      },
      "1:Pump#4":
      {
        "neighbors":
        [
          "0@[Reference)": Pipe#2
        ],
        "players":
        [],
        "output@[Reference)": Pipe#2,
        "hasStoredWater": false,
        "isBroken": false
      }
    ],
    "players":
    [
      "0@[Reference)": Mechanic#1
    ],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
  },
  "carriedPump:Pump#5":
  {
    "neighbors":
    [],
    "players":
    [],
    "hasStoredWater": false,
    "isBroken": false
  }
},
],
}
```

```

"fields":
[
    "0@[Reference)": Cistern#3,
    "1@[Reference)": Pump#4,
    "2@[Reference)": Pipe#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 0,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Cistern#3,
        "2@[Reference)": Pump#4
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}
)
play
place
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [
                    "0:Cistern#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pipe#2
                        ],
                        "players":
                        [],
                        "input@[Reference)": Pipe#2
                    },
                    "1:Pump#4":
                    {
                        "neighbors":
                        [

```

```

        "0@[Reference)": Pipe#2
    ],
    "players":
    [],
    "output@[Reference)": Pipe#2,
    "hasStoredWater": false,
    "isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
},
"carriedPump:Pump#5":
{
    "neighbors":
    [],
    "players":
    [],
    "hasStoredWater": false,
    "isBroken": false
}
},
"fields":
[
    "0@[Reference)": Cistern#3,
    "1@[Reference)": Pump#4,
    "2@[Reference)": Pipe#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 0,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Cistern#3,
        "2@[Reference)": Pump#4
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2
    ]
}

```

## 8.2.16 Szerelő sikeresen felvesz egy pumpát

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy “pumpát felvesz” parancsot, melynek következtében a szerelőhöz kerül egy pumpa. A művelet azért volt sikeres, mert a szerelő ciszternán áll, a ciszternán van pumpa és a játékosnak van még akciótörlő.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív szerelő játékos pumpát vesz fel.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a játékos nem tudja felvenni a pumpát
3. az actionNumber nem csökken

- Bemenet

output

```
load -f tests/map/map16.txt
(
"Game#0":
{
  "players": [
    {
      "0:Mechanic#1": {
        "activeField:Cistern#2": {
          "neighbors": [],
          "players": [
            {
              "0@[Reference)": Mechanic#1
            }
          ],
          "spawnedPump:Pump#3": {
            "neighbors": [],
            "players": [],
            "hasStoredWater": false,
            "isBroken": false
          }
        }
      }
    },
    {
      "fields": [
        {
          "0@[Reference)": Cistern#2
        },
        {
          "saboteurPoints": 0,
          "mechanicPoints": 0,
          "actionNumber": 3,
        }
      ]
    }
  ]
}
```

```

    "activePlayer@[Reference)": Mechanic#1
},
"Timer#4":
{
    "periodics":
    [
        "0@[Reference)": Cistern#2
    ],
    "statefuls":
    []
}
}

)
play
pickup
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [],
                "players":
                [
                    "0@[Reference)": Mechanic#1
                ]
            },
            "carriedPump:Pump#3":
            {
                "neighbors":
                [],
                "players":
                [],
                "hasStoredWater": false,
                "isBroken": false
            }
        }
    ],
    "fields":
    [
        "0@[Reference)": Cistern#2
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 2,
    "activePlayer@[Reference)": Mechanic#1
},
"Timer#4":
{

```

```

"periodics":  

[  

    "0@[Reference)": Cistern#2  

],  

"statefuls":  

[]  

}

```

## 8.2.17 Szerelő nem tud pumpát felvenni

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy “pumpát felvesz” parancsot, de a szerelőhöz nem kerül pumpa a következő okok valamelyike miatt: a szerelő nem ciszternán áll, a szerelő olyan ciszternán áll amin nincs pumpa, a játékosnak nincs több akciót pontja, a játékosnál már van pumpa.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív szerelő játékos pumpát vesz fel, ciszternán van-e pumpa ellenőrzés, szerelőnél van-e pumpa ellenőrzés.

Várható hibahelyek:

- load-ban megadott fájl nem létezik
- a játékos tud felvenni pumpát, pedig nem minden feltétel adott a sikeresz
- az actionNumber csökken
- a spawned pump eltünik a ciszterntáról

- Bemenet

```

output  

load -f tests/map/map17.txt  

()  

"Game#0":  

{  

    "players":  

[  

    "0:Mechanic#1":  

    {  

        "activeField:Cistern#2":  

        {  

            "neighbors":  

[],  

            "players":  

[  

                "0@[Reference)": Mechanic#1  

],  

            "spawnedPump:Pump#3":  

            {  

                "neighbors":  

[],  

                "players":  

[],  

                "hasStoredWater": false,  

                "isBroken": false  

            }  

},  

        "carriedPump:Pump#4":  

}
}
```

```

        {
            "neighbors": [],
            "players": [],
            "hasStoredWater": false,
            "isBroken": false
        }
    },
    "fields": [
        {
            "0@[Reference)": Cistern#2
        },
        "saboteurPoints": 0,
        "mechanicPoints": 0,
        "actionNumber": 3,
        "activePlayer@[Reference)": Mechanic#1
    },
    "Timer#5": {
        "periodics": [
            {
                "0@[Reference)": Cistern#2
            },
            "statefuls": []
        }
    }
)
play
pickup
    • Elvárt kimenet
"Game#0": {
    "players": [
        {
            "0:Mechanic#1": {
                "activeField:Cistern#2": {
                    "neighbors": [],
                    "players": [
                        {
                            "0@[Reference)": Mechanic#1
                        }
                    ],
                    "spawnedPump:Pump#3": {
                        "neighbors": []
                    }
                }
            }
        }
    ]
}

```

```

        ],
        "players": [
        ],
        "hasStoredWater": false,
        "isBroken": false
    }
},
"carriedPump:Pump#4":
{
    "neighbors": [
    ],
    "players": [
    ],
    "hasStoredWater": false,
    "isBroken": false
}
],
"fields": [
    "0@[Reference)": Cistern#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#5":
{
    "periodics": [
        "0@[Reference)": Cistern#2
    ],
    "statefuls": []
}
}

```

### 8.2.18 Játékos sikeresen átcsatlakoztat egy csővéget

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet átcsatlakoztat parancsot, melynek következtében egy cső vége más mezőhöz fog kapcsolódni. A művelet azért sikeres, mert az új végként megadott mező nem egyezik meg az eddigi véggel, valamint a játékosnak van még akciót pontja.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos csövet köt át.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a movePipe valamely megadott paramétere nem létezik
3. a játékos nem tudja átállítani a csővéget, pedig minden feltétel adott hozzá

- **Bemenetoutput**

```

load -f tests/map/map18.txt
(
"Game#0":
{
  "players":
  [
    "0:Mechanic#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [],
        "players":
        [],
        "hasStoredWater": false,
        "isBroken": false
      }
    }
  ],
  "fields":
  [
    "0:Pump#3":
    {
      "neighbors":
      [
        "0:Pipe#4":
        {
          "neighbors":
          [
            "0@[Reference)": Pump#3,
            "1:Pump#5":
            {
              "neighbors":
              [
                "1@[Reference)": Pipe#4,
              ],
              "players":
              [],
              "output@[Reference)": Pipe#4,
              "hasStoredWater": false,
              "isBroken": false
            }
          ],
          "players":
          [],
          "oldWaterState": false,
          "breakProofCounter": 0,
          "slipperyCounter": 0,
          "stickyCounter": 0,
        }
      ]
    }
  ]
}
)

```

```

        "newWaterState": false,
        "isBroken": false
    },
],
"players":
[
    {
        "0@[Reference)": Mechanic#1
    },
    "input@[Reference)": Pipe#4,
    "hasStoredWater": false,
    "isBroken": false
},
{
    "1@[Reference)": Pump#5,
    "2@[Reference)": Pump#2,
    "3@[Reference)": Pipe#4
},
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#3,
        "1@[Reference)": Pump#5,
        "2@[Reference)": Pump#2,
        "3@[Reference)": Pipe#4
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#4
    ]
}
}

)
play
movePipe -p Pipe#4 -oe Pump#3 -ne Pump#2
• Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [

```

```

    "0:Pipe#3":
    {
        "neighbors":
        [
            "0:Pump#4":
            {
                "neighbors":
                [
                    "0@[Reference)": Pipe#3,
                ],
                "players":
                [
                    "0@[Reference]":
                ],
                "input@[Reference)": Pipe#3,
                "hasStoredWater": false,
                "isBroken": false
            },
            "1@[Reference)": Pump#2
        ],
        "players":
        [],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
},
"players":
[],
"hasStoredWater": false,
"isBroken": false
}
},
"fields":
[
    "0@[Reference)": Pump#4,
    "1:Pump#5":
    {
        "neighbors":
        [
            "1@[Reference)": Pipe#3
        ],
        "players":
        [],
        "hasStoredWater": false,
    }
]
]

```

```

        "isBroken": false
    },
    "2@[Reference)": Pump#2,
    "3@[Reference)": Pipe#3
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 2,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#4,
        "1@[Reference)": Pump#5,
        "2@[Reference)": Pump#2,
        "3@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}

```

## 8.2.19 Játékos nem tud átkötni egy csővéget

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy csövet átcsatlakoztat parancsot, de a cső vége nem fog más mezőhöz kapcsolódni a következő okok valamelyike miatt: a megadott új vég megegyezik az eddigivel, a megadott cső nem szomszédos a régi vég paraméterként átadott mezővel, vagy a játékosnak nincs több akciópontja.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos csövet köt át, új vég = régi vég ellenőrzése.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a movePipe valamely megadott paramétere nem létezik
3. a játékos át tudja állítani a csővéget, pedig nem minden feltétel adott hozzá

- Bemenet

```

output
load -f tests/map/map19.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":

```

```
[
    "0:Pipe#3":
    {
        "neighbors":
        [
            "0@[Reference)": Pump#2,
            "1:Cistern#4":
            {
                "neighbors":
                [
                    "0@[Reference)": Pipe#3
                ],
                "players":
                [],
                "input@[Reference)": Pipe#3
            }
        ],
        "players":
        [],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    },
    ],
    "players":
    [
        "0@[Reference)": Mechanic#1
    ],
    "input@[Reference)": Pipe#3,
    "hasStoredWater": false,
    "isBroken": false
},
],
"fields":
[
    "0@[Reference)": Pump#2,
    "1@[Reference)": Cistern#4,
    "2:Source#5":
    {
        "neighbors":
        [],
        "players":
        []
    },
    "3@[Reference)": Pipe#3
],
]
```

```

"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#2,
        "1@[Reference)": Cistern#4,
        "2@[Reference)": Source#5,
        "3@[Reference)": Pipe#3
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}

)
play
movePipe -p Pipe#2 -oe Cistern#4 -ne Source#5
● Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pump#2,
                            "1:Cistern#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                                "players":
                                [],
                                "input@[Reference)": Pipe#3
                            }
                        ]
                    }
                ]
            }
        }
    ]
}

```

```

        "players":  

        []],  

        "oldWaterState": false,  

        "breakProofCounter": 0,  

        "slipperyCounter": 0,  

        "stickyCounter": 0,  

        "newWaterState": false,  

        "isBroken": false  

    },  

    "2@[Reference)": Pipe#3  

],  

"players":  

[  

    "0@[Reference)": Mechanic#1  

],  

"input@[Reference)": Pipe#3,  

"hasStoredWater": false,  

"isBroken": false  

}  

}  

],  

"fields":  

[  

    "0@[Reference)": Pump#2,  

    "1@[Reference)": Cistern#4,  

    "2:Source#5":  

    {  

        "neighbors":  

        [],  

        "players":  

        []  

    },  

    "3@[Reference)": Pipe#3  

],  

"saboteurPoints": 0,  

"mechanicPoints": 0,  

"actionNumber": 3,  

"activePlayer@[Reference)": Mechanic#1  

},  

"Timer#6":  

{  

    "periodics":  

[  

    "0@[Reference)": Pump#2,  

    "1@[Reference)": Cistern#4,  

    "2@[Reference)": Source#5,  

    "3@[Reference)": Pipe#3  

],  

"statefuls":  

[  

]

```

```

        "0@[Reference)": Pipe#3
    ]
}

```

## 8.2.20 Játékos sikeresen átállítja egy pumpa be- és kimenetét

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát átállít parancsot, melynek következtében a pumpa, amelyen a játékos áll, más bemenetről fogad és más kimenetre küld vizet, mint eddig. A művelet azért sikeres, mert a játékos valóban pumpán áll, az új be- és kimenet szomszédai a pumpának, az új be- és kimenet nem ugyanaz a cső és a játékosnak van még akciótömbje.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos pumpát állít át.

Várható hibahelyek:

- load-ban megadott fájl nem létezik
- a changePumpDirection valamely megadott paramétere nem létezik
- a játékos nem tudja átállítani a pumpát, pedig minden feltétel adott hozzá

- Bemenet

```

output
load -f tests/map/map20.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pump#2
                        ],
                        "players":
                        [],
                        "oldWaterState": false,
                        "breakProofCounter": 0,
                        "slipperyCounter": 0,
                        "stickyCounter": 0,
                        "newWaterState": false,
                        "isBroken": false
                    },
                    "1:Pipe#4":
                    {
                        "neighbors":
                        [

```

```

        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"2@[Reference)": Pipe#3,
"3@[Reference)": Pipe#4,
"4:Pipe#5":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3,
"output@[Reference)": Pipe#4,
"hasStoredWater": false,
"isBroken": false
}
},
"fields":
[
    "0@[Reference)": Pump#2,
    "1@[Reference)": Pipe#3,
    "2@[Reference)": Pipe#4,
    "3@[Reference)": Pipe#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,

```

```

    "activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#2,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#4,
        "3@[Reference)": Pipe#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#4,
        "2@[Reference)": Pipe#5
    ]
}

)
play
changePumpDirection -ip Pipe#3 -op Pipe#5
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pump#2
                        ],
                        "players":
                        [],
                        "oldWaterState": false,
                        "breakProofCounter": 0,
                        "slipperyCounter": 0,
                        "stickyCounter": 0,
                        "newWaterState": false,
                        "isBroken": false
                    },
                    "1:Pipe#4":
                    {
                }
            }
        }
    ]
}

```

```

    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"2@[Reference)": Pipe#3,
"3@[Reference)": Pipe#4,
"4:Pipe#5":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3,
"output@[Reference)": Pipe#5,
"hasStoredWater": false,
"isBroken": false
}
},
"fields":
[
    "0@[Reference)": Pump#2,
    "1@[Reference)": Pipe#3,
    "2@[Reference)": Pipe#4,
    "3@[Reference)": Pipe#5
],
"saboteurPoints": 0,

```

```

    "mechanicPoints": 0,
    "actionNumber": 2,
    "activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#2,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#4,
        "3@[Reference)": Pipe#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#4,
        "2@[Reference)": Pipe#5
    ]
}

```

### 8.2.21 Játékos nem tudja átállítani egy pumpa be- és kimenetét

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy pumpát átállít parancsot, de a művelet sikertelen a következő okok valamelyike miatt: a játékos nem pumpán áll, a megadott új be- és kimenet valamelyike (vagy mindenek) nem szomszédos a pumpával, amin a játékos áll, a megadott új be- és kimenet ugyanaz a cső, a játékosnak nincs több akciót pontja.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív játékos pumpát állít át, szomszédos-e a megadott mező ellenőrzés, megegyezik-e a két megadott mező ellenőrzés.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. a changePumpDirection valamely megadott paramétere nem létezik
3. a játékos át tudja átállítani a pumpát, pedig nem minden feltétel adott hozzá

- **Bemenet**

```

output
load -f tests/map/map21.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {

```

```

    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"1:Pipe#4":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"2@[Reference)": Pipe#3,
"3@[Reference)": Pipe#4,
"4:Pipe#5":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
]

```

```

        "input@[Reference)": Pipe#3,
        "output@[Reference)": Pipe#4,
        "hasStoredWater": false,
        "isBroken": false
    }
}
],
"fields":
[
    "0@[Reference)": Pump#2,
    "1@[Reference)": Pipe#3,
    "2@[Reference)": Pipe#4,
    "3@[Reference)": Pipe#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#2,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#4,
        "3@[Reference)": Pipe#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#4,
        "2@[Reference)": Pipe#5
    ]
}
)
play
changePumpDirection -ip Pipe#3 -op Pipe#5
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":

```

```
{
    "neighbors": [
        "0@[Reference)": Pump#2
    ],
    "players": [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"1:Pipe#4":
{
    "neighbors": [
        "0@[Reference)": Pump#2
    ],
    "players": [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"2@[Reference)": Pipe#3,
"3@[Reference)": Pipe#4,
"4:Pipe#5":
{
    "neighbors": [
        "0@[Reference)": Pump#2
    ],
    "players": [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
}
],
"players": [
    "0@[Reference)": Mechanic#1
]
```

```

        ],
        "input@[Reference)": Pipe#3,
        "output@[Reference)": Pipe#4,
        "hasStoredWater": false,
        "isBroken": false
    }
},
],
"fields":
[
    "0@[Reference)": Pump#2,
    "1@[Reference)": Pipe#3,
    "2@[Reference)": Pipe#4,
    "3@[Reference)": Pipe#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pump#2,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#4,
        "3@[Reference)": Pipe#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#4,
        "2@[Reference)": Pipe#5
    ]
}
}

```

## 8.2.22 (összetett) Lépés + csúszás + pumpa átállítás

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: léptető, pumpát átállító. A léptető parancs következtében a játékos rálép a célként megadott mezőre, ami egy csúszós cső, ezért átcsúszik egy pumpára. A pumpára érve a második parancsal átállítja a pumpa irányát, feltéve, hogy a megadott új be- és kimenet érvényes és a játékosnak van még akciótponja.

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos pumpát átállít, szomszédos-e a megadott mező ellenőrzés, megegyezik-e a két megadott mező ellenőrzés, csúszós cső viselkedése, játékos lépése, random választás (csúszós csőről hova menjen).

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a játékos nem tud átlépni a céljára pedig kéne tudnia
4. nem csúszik át az új mezőre - az a régi amelyikben víz van
5. a pumpát nem tudja átállítani pedig kéne tudnia
6. a changePumpDirection valamelyik paramétere nem létezik

- **Bemenet**

```

output
random //kikapcsoljuk
load -f tests/map/map22.txt
(
"Game#0":
{
  "players":
  [
    "0:Mechanic#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [
          "0:Pipe#3":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2
            ],
            "players":
            [],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
          },
          "1:Pipe#4":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2,
              "1:Pump#5":
              {
                "neighbors":
                [
                  "0@[Reference)": Pipe#4,
                  "1:Pipe#6":
                  {
                    "neighbors":
                    [

```

```

    "0@[Reference)": Pump#5
        ],
        "players":
        [],
        "oldWaterState":
        false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
],
"players":
[],
"input@[Reference)": Pipe#4,
"output@[Reference)": Pipe#6,
"hasStoredWater": false,
"isBroken": false
}
],
"players":
[],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 4,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3,
"output@[Reference)": Pipe#4,
"hasStoredWater": true,
"isBroken": false
}
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pipe#4,
]

```

```

    "2@[Reference)": Pipe#6,
    "3@[Reference)": Pump#2,
    "4@[Reference)": Pump#5
  ],
  "saboteurPoints": 0,
  "mechanicPoints": 0,
  "actionNumber": 3,
  "activePlayer@[Reference)": Mechanic#1
},
"Timer#7":
{
  "periodics":
  [
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pipe#4,
    "2@[Reference)": Pipe#6,
    "3@[Reference)": Pump#2,
    "4@[Reference)": Pump#5
  ],
  "statefuls":
  [
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pipe#4,
    "2@[Reference)": Pipe#6
  ]
}
)
play
move -f Pipe#4 //csusszon az új oldalra
changePumpDirection -ip Pipe#6 -op Pipe#4
  • Elvárt kimenet
"Game#0":
{
  "players":
  [
    "0:Mechanic#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [
          "0:Pipe#3":
          {
            "neighbors":
            [
              "0:Pump#4":
              {
                "neighbors":
                [
                  "0:Pipe#5":
                  {
                    "neighbors":
                    [

```

```
[

    "0@[Reference)": Pump#4
        ],
        "players": []
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    },
    "1@[Reference)": Pipe#3
        ],
        "players": []
        ],
        "input@[Reference)": Pipe#5,
        "output@[Reference)": Pipe#3,
        "hasStoredWater": true,
        "isBroken": false
    },
    "1@[Reference)": Pump#2
        ],
        "players": []
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 4,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    },
    "1:Pipe#6":
    {
        "neighbors":
        [
            "0@[Reference)": Pump#2
        ],
        "players": []
        ],
        "oldWaterState": false,
        "breakProofCounter": 0,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
},
"players": []
]
```

```

        "0@[Reference)": Mechanic#1
    ],
    "input@[Reference)": Pipe#6,
    "output@[Reference)": Pipe#3,
    "hasStoredWater": false,
    "isBroken": false
}
],
"fields":
[
    "0@[Reference)": Pipe#5,
    "1@[Reference)": Pipe#3,
    "2@[Reference)": Pipe#6,
    "3@[Reference)": Pump#4,
    "4@[Reference)": Pump#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 1,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#7":
{
    "periodics":
    [
        "0@[Reference)": Pipe#5,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#6,
        "3@[Reference)": Pump#4,
        "4@[Reference)": Pump#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#5,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#6
    ]
}
}

```

## 8.2.23 (összetett) Játékos lyukaszt, csúszóssá tesz majd ellép

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: csövet kilyukaszt, csúszóssá tesz, lép. Az első parancs hatására a játékos kilyukasztja a csövet, amin áll (feltéve, hogy csövön áll és az lyukasztható állapotban van), a második következtében csúszóssá teszi a csövet, amin áll (ezt mindenképpen megteheti, ha van akciót pontja), végül ellép egy megadott pályaelemre (ezt biztosan megteheti, ha van elég akciót pontja, hiszen csőnek nem lehet cső a szomszédja).

- Ellenőrzött funkcionalitás, várható hibahelyek

Az aktív játékos csövet lyukaszt, lyukasztható állapot ellenőrzése, csövet csúszóssá tesz, lépés.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik

2. move-ban megadott mező nem létezik
3. a cső nem lyukad ki
4. a cső slipperyCounterje egy 5-től eltérő szám
5. a játékos nem a pumpán áll
6. a játékosnak nem 0 az actionNumberje

- **Bemenet**

```

output
load -f tests/map/map23.txt
(
"Game#0":
{
  "players":
  [
    "0:Saboteur#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [
          "0:Pipe#3":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2
            ],
            "players":
            [],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
          }
        ],
        "players":
        [
          "0@[Reference)": Saboteur#1
        ],
        "input@[Reference)": Pipe#3,
        "hasStoredWater": false,
        "isBroken": false
      }
    }
  ],
  "fields":
  [
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pump#2
  ],
}
)

```

```

    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Saboteur#1
},
"Timer#4":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
)
play
breakField
makeSlippery
move -f Pump#2
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Saboteur#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pump#2
                        ],
                        "players":
                        [],
                        "oldWaterState": false,
                        "breakProofCounter": 0,
                        "slipperyCounter": 5,
                        "stickyCounter": 0,
                        "newWaterState": false,
                        "isBroken": true
                    }
                ],
                "players":
                [
                    "0@[Reference)": Saboteur#1
                ],
            }
        }
    ]
}

```

```

        "input@[Reference)": Pipe#3,
        "hasStoredWater": false,
        "isBroken": false
    }
},
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pump#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 0,
"activePlayer@[Reference)": Saboteur#1
},
"Timer#4":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
}

```

## 8.2.24 (összetett) Szerelő felvesz egy pumpát, lép, majd leteszí

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: pumpát felvesz, lép, pumpát letesz. Az első hatására a szerelő játékos felvesz egy pumpát a ciszternáról, amin áll (ha azon áll és ott van pumpa), a második parancs hatására ellép a megadott csőre (feltéve, hogy ott nem áll más, ciszternának csak cső lehet a szomszédja), végül leteszí a pumpáját a csőre, amin áll (ha maradt még akciópontja).

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az aktív játékos pumpát vesz fel, van-e pumpa a ciszternán ellenőrzés, van-e pumpa a játékosnál ellenőrzés, lépés, csövön állnak-e ellenőrzés, pumpát letesz.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a játékos nem tud pumpát felvenni pedig kéne tudnia
4. a játékos nem tud pumpát letenni pedig minden feltétel adott hozzá
5. a játékos nem tud lépni pedig minden feltétel adott hozzá

- **Bemenet**

```

output
load -f tests/map/map24.txt
(
"Game#0":
{
    "players":
```

```
[
  "0:Mechanic#1":
  {
    "activeField:Cistern#2":
    {
      "neighbors":
      [
        "0:Pipe#3":
        {
          "neighbors":
          [
            "0@[Reference)": Cistern#2,
            "1:Pump#4":
            {
              "neighbors":
              [
                "0@[Reference)": Pipe#3
              ],
              "players":
              [],
              "output@[Reference)": Pipe#3,
              "hasStoredWater": false,
              "isBroken": false
            }
          ],
          "players":
          [],
          "oldWaterState": false,
          "breakProofCounter": 0,
          "slipperyCounter": 0,
          "stickyCounter": 0,
          "newWaterState": false,
          "isBroken": false
        }
      ],
      "players":
      [
        "0@[Reference)": Mechanic#1
      ],
      "input@[Reference)": Pipe#3,
      "spawnedPump:Pump#5":
      {
        "neighbors":
        [],
        "players":
        [],
        "hasStoredWater": false,
        "isBroken": false
      }
    }
  }
]
```

```

        }
    ],
    "fields": [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Cistern#2
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Mechanic#1
},
"Timer#6":
{
    "periodics": [
        "0@[Reference)": Cistern#2,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pump#4
    ],
    "statefuls": [
        "0@[Reference)": Pipe#3
    ]
}
)
play
pickup
move -f Pipe#3
place


- Elvárt kimenet


"Game#0":
{
    "players": [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pump#2,
                            "1:Cistern#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ]
                            }
                        ]
                    }
                ]
            }
        }
    ]
}

```

```

        ],
        "players": []
    }
},
"players": [],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
},
"1:Pipe#5":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2,
        "1:Pump#6":
        {
            "neighbors":
            [
                "0@[Reference)": Pipe#5
            ],
            "players": []
        },
        "hasStoredWater": false,
        "isBroken": false
    }
},
"players": [],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players": [
    "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3,
"output@[Reference)": Pipe#5,
"hasStoredWater": false,
"isBroken": false
}
},
"fields":
[
    "0@[Reference)": Pump#6,

```

```

    "1@[Reference)": Cistern#4,
    "2@[Reference)": Pump#2,
    "3@[Reference)": Pipe#3,
    "4@[Reference)": Pipe#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 0,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#7":
{
    "periodics":
    [
        "0@[Reference)": Cistern#4,
        "1@[Reference)": Pump#6,
        "2@[Reference)": Pump#2,
        "3@[Reference)": Pipe#3,
        "4@[Reference)": Pipe#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#5
    ]
}
}

```

## 8.2.25 Hárrom játékos, hat lépés, hat mező #1

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: csövet lyukaszt, kört befejez, pumpát átállít, kört befejez, pumpát átállít, lép. Az első két parancsot (csövet lyukaszt, kört befejez) az első játékos (szabotőr) hajtja végre: kilyukasztja a csövet amin áll, majd befejezi a körét. A következő két parancsot (pumpát átállít, kört befejez) a második játékos (szerelő) hajtja végre: átállítja a kilyukasztott cső forráspumpájának be- és kimenetét, majd befejezi a körét. Az utolsó két parancsot (pumpát átállít, lép) a harmadik játékos (szabotőr) hajtja végre: átállítja a fentebb tárgyalt cső célpumpáját úgy, hogy a cső ne a bemenete legyen, hanem a kimenete (bemenetnek pedig választ egy másik csövet), majd ellép máshova. Mindhárrom játékos veszít 2-2 akciót pontot. (tegyük fel, hogy a végrehajtott műveletek sikeresek és volt elég akciót pontja minden hárrom játékosnak a végrehajtásukhoz)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Aktív játékos csövet lyukaszt, kört befejez, pumpát átállít, lép.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a cső nem lyukad ki
4. a pumpák rossz input-outputtal rendelkeznek
5. a harmadik játékos rossz helyen áll
6. az első/második játékos nem tudja befejezni a körét
7. valamelyik changePumpDirection valamely paramétere nem létezik.
8. az actionNumber nem 1

- **Bemenet**

```

output
load -f tests/map/map25.txt
(
"Game#0":
{
  "players":
  [
    "0:Mechanic#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [
          "0:Pipe#3":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2
            ],
            "players":
            [],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
          },
          "1:Pipe#4":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2,
              "1:Pump#5":
              {
                "neighbors":
                [
                  "0@[Reference)": Pipe#4,
                  "1:Pipe#6":
                  {
                    "neighbors":
                    [
                      "0@[Reference)": Pipe#4,
                      "1:Pipe#6":
                      {
                        "neighbors":
                        [
                          "0@[Reference)": Pump#5
                        ],
                        "players":
                        [],
                        "oldWaterState": false,
                      }
                    ]
                  }
                ]
              }
            ]
          }
        ]
      }
    }
  ]
}

```

```

"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players":
[
    "0:Saboteur#7":
    {
        "activeField@[Reference)": Pump#5
        }
    ],
    "input@[Reference)": Pipe#4,
    "output@[Reference)": Pipe#6,
    "hasStoredWater": false,
    "isBroken": false
}
],
"players":
[
    "0:Saboteur#8":
    {
        "activeField@[Reference)": Pipe#4
        }
    ],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": false
},
"2:Pipe#9":
{
    "neighbors":
    [
        "0@[Reference)": Pump#2
    ],
    "players":
    [],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
}

```

```

        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
],
"players":
[
    {
        "0@[Reference)": Mechanic#1
    },
    {
        "input@[Reference)": Pipe#3,
        "output@[Reference)": Pipe#4,
        "hasStoredWater": false,
        "isBroken": false
    }
},
{
    "1@[Reference)": Saboteur#7,
    "2@[Reference)": Saboteur#8
},
"fields":
[
    "0@[Reference)": Pipe#4,
    "1@[Reference)": Pump#2,
    "2@[Reference)": Pipe#6,
    "3@[Reference)": Pump#5,
    "4@[Reference)": Pipe#3,
    "5@[Reference)": Pipe#9
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#8
},
"Timer#10":
{
    "periodics":
    [
        "0@[Reference)": Pipe#4,
        "1@[Reference)": Pipe#6,
        "2@[Reference)": Pipe#3,
        "3@[Reference)": Pipe#9,
        "4@[Reference)": Pump#2,
        "5@[Reference)": Pump#5
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#4,
        "1@[Reference)": Pipe#6,
        "2@[Reference)": Pipe#3,
        "3@[Reference)": Pipe#9
    ]
}

```

```

}
)
play
breakField //Pipe#4
endTurn
changePumpDirection -ip Pipe#3 -op Pipe#9 //Pump#2
endTurn
changePumpDirection -ip Pipe#6 -op Pipe#4 //Pump#5
move -f Pipe#6
● Elvárt kimenet
"Game#0":
{
  "players": [
    "0:Saboteur#1": {
      "activeField:Pipe#2": {
        "neighbors": [
          "0:Pump#3": {
            "neighbors": [
              "0:Pipe#4": {
                "neighbors": [
                  "0@[Reference)": Pump#3
                ],
                "players": [],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
              },
              "1@[Reference)": Pipe#2,
              "2:Pipe#5": {
                "neighbors": [
                  "0@[Reference)": Pump#3
                ],
                "players": [],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
              }
            ]
          }
        ]
      }
    }
  ]
}

```

```

        }
    ],
    "players": [
        {
            "0:Mechanic#6": {
                "activeField@[Reference)": Pump#3
            }
        },
        {
            "input@[Reference)": Pipe#4,
            "output@[Reference)": Pipe#5,
            "hasStoredWater": false,
            "isBroken": false
        },
        "1:Pump#7": {
            "neighbors": [
                {
                    "0@[Reference)": Pipe#2,
                    "1:Pipe#8": {
                        "neighbors": [
                            {
                                "0@[Reference)": Pump#7
                            },
                            "players": [
                                {
                                    "0:Saboteur#9": {
                                        ...
                                    }
                                }
                            ],
                            "oldWaterState": false,
                            "breakProofCounter": 0,
                            "slipperyCounter": 0,
                            "stickyCounter": 0,
                            "newWaterState": false,
                            "isBroken": false
                        }
                    }
                ],
                "players": [],
                "input@[Reference)": Pipe#8,
                "output@[Reference)": Pipe#2,
                "hasStoredWater": false,
                "isBroken": false
            }
        },
        "players": [
            {
                "0@[Reference)": Saboteur#1
            },
            "oldWaterState": false,
            "breakProofCounter": 0,
        ]
    ]
}

```

```

        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": true
    }
},
"1@[Reference)": Mechanic#6,
"2@[Reference)": Saboteur#9
],
"fields":
[
    "0@[Reference)": Pipe#2,
    "1@[Reference)": Pump#3,
    "2@[Reference)": Pipe#8,
    "3@[Reference)": Pump#7,
    "4@[Reference)": Pipe#4,
    "5@[Reference)": Pipe#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 1,
"activePlayer@[Reference)": Saboteur#9
},
"Timer#10":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Pipe#8,
        "2@[Reference)": Pipe#4,
        "3@[Reference)": Pipe#5,
        "4@[Reference)": Pump#3,
        "5@[Reference)": Pump#7
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Pipe#8,
        "2@[Reference)": Pipe#4,
        "3@[Reference)": Pipe#5
    ]
}
}

```

### 8.2.26 Három játékos, hat lépés, hat mező #2

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: lép, csövet lyukaszt, kört befejez, lép, pumpát javít, kört befejez, csövet átköt, lép. Az első két parancsot (lép, csövet lyukaszt) az első játékos (szabotőr) hajtja végre, előbbi következetében kilyukasztja a csövet, amin áll, utóbbi következetében befejezi a körét. A következő három parancsot (lép, pumpát javít, kört befejez) a második játékos hajtja végre. Az első hatására rálép egy pumpára, a második hatására megjavítja azt, majd befejezi a körét. Az utolsó két parancsot (csövet átköt, lép) a harmadik játékos (szerelő) hajtja végre, ezek

eredményeként megváltoztatja egy cső végeit, majd ellép egy megadott mezőre. (A tesztelés során feltesszük, hogy a műveletek elvégzése a szabályok alapján lehetséges.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Aktív játékos lép, csövet lyukaszt, pumpát javít, csövet átállít, kört befejez.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. az első játékos nem tud csövet lyukasztani pedig kéne tudnia
4. a második játékos nem tud pumpát javítani, pedig kéne tudnia
5. a második játékos nem tud lépni, pedig kéne tudnia
6. a harmadik játékos nem tud csövet átkötni, pedig kéne tudnia

- **Bemenet**

```
output
load -f tests/map/map26.txt
(
"Game#0":
{
  "players":
  [
    "0:Saboteur#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [
          "0:Pipe#3":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2,
              "1:Pump#4":
              {
                "neighbors":
                [
                  "0@[Reference)": Pipe#3,
                  "1:Pipe#5":
                  {
                    "neighbors":
                    [
                      "0@[Reference)": Pipe#5
                    ],
                  }
                ]
              }
            ]
          }
        ]
      }
    }
  ]
}
"0@[Reference)": Pump#4,
"1:Pump#6":
{
  "neighbors":
  [
    "0@[Reference)": Pipe#5
  ],
}
],
```

```

"players": [],
  ],
  "input@[Reference)": Pipe#5,
  "hasStoredWater": false,
  "isBroken": false
    },
  ],
  "players": [
    {
      "0:Mechanic#7":
        {
          "activeField@[Reference)": Pipe#5
            },
          ],
          "oldWaterState": false,
          "breakProofCounter": 0,
          "slipperyCounter": 0,
          "stickyCounter": 0,
          "newWaterState": false,
          "isBroken": false
        },
      "2:Pipe#8":
        {
          "neighbors": [
            {
              "0@[Reference)": Pump#4,
              "1:Pump#9":
                {
                  "neighbors": [
                    {
                      "0@[Reference)": Pipe#8
                        ],
                      ],
                    "players": [
                      [

```

```

    "0:Mechanic#10":
    {
        "activeField@[Reference)": Pump#9
    }
],
"input@[Reference)": Pipe#8,
"hasStoredWater": false,
"isBroken": false
}
],
"players": [],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players": [],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players": []
}
],
"players": []
}
]

```

```

        [
            "0@[Reference)": Saboteur#1
        ],
        "output@[Reference)": Pipe#3,
        "hasStoredWater": false,
        "isBroken": false
    }
},
"1@[Reference)": Mechanic#7,
"2@[Reference)": Mechanic#10
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pump#2,
    "2@[Reference)": Pipe#5,
    "3@[Reference)": Pipe#8,
    "4@[Reference)": Pump#4,
    "5@[Reference)": Pump#6,
    "6@[Reference)": Pump#9
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#1
},
"Timer#11":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2,
        "2@[Reference)": Pipe#5,
        "3@[Reference)": Pipe#8,
        "4@[Reference)": Pump#4,
        "5@[Reference)": Pump#6,
        "6@[Reference)": Pump#9
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#8,
        "2@[Reference)": Pipe#5
    ]
}
)
play
breakField //Pipe#3
endTurn
move -f Pump#4

```

```

repairField
endTurn
movePipe -p Pipe#5 -oe Pump#6 -ne Pump#9
move -f Pipe#5
● Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Saboteur#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [
                    "0:Pump#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pipe#2
                        ],
                        "players":
                        [],
                        "output@[Reference)": Pipe#2,
                        "hasStoredWater": false,
                        "isBroken": false
                    },
                    "1:Pump#4":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pipe#2,
                            "1:Pipe#5":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pump#4,
                                    "1:Pump#6":
                                    {
                                        "neighbors":
                                        [
                                            "0@[Reference)": Pump#4,
                                            "1:Pipe#7":
                                            {
                                                "1:Pipe#7":
                                                {
                                                    "neighbors":
                                                    [
                                                        "0:Pump#8":
                                                        {
                                                            "neighbors":
                                                            [
                                                                "0:Pump#8":
                                                                {
                                                                    "neighbors":
                                                                    [
                                                                ]
                                                            ]
                                                        ]
                                                    ]
                                                ]
                                            ]
                                        ]
                                    ]
                                ]
                            ]
                        ]
                    }
                ]
            ]
        }
    ]
}

```

```
[
    "0@[Reference)": Pipe#7
],
"players":
[],
"input@[Reference)": Pipe#7,
"hasStoredWater": false,
"isBroken": false
},
"1@[Reference)": Pump#6
],
"players":
[
"0:Mechanic#9":
{
    "activeField@[Reference)": Pipe#7
}
],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players":
[],
"input@[Reference)": Pipe#5,
"hasStoredWater": false,
"isBroken": false
}
}
```

```

        ],
        "players": [
            ],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 0,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
        }
    ],
    "players": [
        {
            "0:Mechanic#10": {
                "activeField@[Reference)": Pump#4
            }
        },
        {
            "input@[Reference)": Pipe#2,
            "hasStoredWater": false,
            "isBroken": false
        }
    ],
    "players": [
        {
            "0@[Reference)": Saboteur#1
        }
    ],
    "oldWaterState": false,
    "breakProofCounter": 0,
    "slipperyCounter": 0,
    "stickyCounter": 0,
    "newWaterState": false,
    "isBroken": true
},
{
    "1@[Reference)": Mechanic#10,
    "2@[Reference)": Mechanic#9
},
"fields": [
    "0@[Reference)": Pipe#2,
    "1@[Reference)": Pump#3,
    "2@[Reference)": Pipe#7,
    "3@[Reference)": Pipe#5,
    "4@[Reference)": Pump#4,
    "5@[Reference)": Pump#8,
    "6@[Reference)": Pump#6
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 1,
"activePlayer@[Reference)": Mechanic#9
},
"Timer#11": {

```

```

"periodics":
[
    "0@[Reference)": Pipe#2,
    "1@[Reference)": Pump#3,
    "2@[Reference)": Pipe#7,
    "3@[Reference)": Pipe#5,
    "4@[Reference)": Pump#4,
    "5@[Reference)": Pump#8,
    "6@[Reference)": Pump#6
],
"statefuls":
[
    "0@[Reference)": Pipe#2,
    "1@[Reference)": Pipe#5,
    "2@[Reference)": Pipe#7
]
}

```

## 8.2.27 Három játékos, hat lépés, hat mező #3

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsok sorozatát: csövet átköt, kört befejez, lép, pumpát átállít, kört befejez, csövet átköt. Az első két parancsot (csövet átköt, kört befejez) az első játékos (szerelő) hajtja végre, melyek következtében átállítja egy cső végeit, majd átadja a kört a következő játékosnak. A következő három parancsot (lép, pumpát átállít, kört befejez) a második játékos (szerelő) végzi, melyek következtében rálép egy pumpára, átállítja annak ki- és bemenetét, végül befejezi a körét. Az utolsó parancsot (csövet átköt) a harmadik játékos (szerelő) hajtja végre, eredményeként átköti egy cső végeit.(A tesztelés során feltesszük, hogy a műveletek elvégzése a szabályok alapján lehetséges.)

- Ellenőrzött funkcionalitás, várható hibahelyek

Aktív játékos csövet átköt, kört befejez, lép, pumpát átállít.

Várható hibahelyek:

- load-ban megadott fájl nem létezik
- move-ban megadott mező nem létezik
- az első játékos nem tud csövet átkötni, pedig kéne tudnia
- a második játékos nem tud pumpát átállítani, pedig kéne tudnia
- a második játékos nem tud lépni, pedig kéne tudnia
- a harmadik játékos nem tud csövet átkötni, pedig kéne tudnia

- Bemenet

```

output
load -f tests/map/map27.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":

```

```

        "0:Pipe#3":
        {
            "neighbors":
            [
                "0@[Reference)": Pump#2
            ],
            "players":
            [
                "0:Mechanic#4":
                {
                    "activeField@[Reference]":
                    {}
                },
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
            },
            "1:Pipe#5":
            {
                "neighbors":
                [
                    "0@[Reference)": Pump#2,
                    "1:Pump#6":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pipe#5,
                            "1:Pipe#7":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pump#6,
                                    "1:Cistern#8":
                                    {
                                        "neighbors":
                                        [
                                            "0@[Reference)": Pipe#7
                                        ],
                                        "players":
                                        [
                                            "0@[Reference)": Pipe#7
                                        ]
                                    }
                                ]
                            }
                        ]
                    }
                ]
            }
        }
    ]
}

```

```

    "0:Mechanic#9": {
      "activeField@[Reference)": Cistern#8
    },
    "input@[Reference)": Pipe#7
  },
  ],
  "players": [],
  "oldWaterState": false,
  "breakProofCounter": 0,
  "slipperyCounter": 0,
  "stickyCounter": 0,
  "newWaterState": false,
  "isBroken": false
},
],
"players": [],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
},
],
"players": [
  "0@[Reference)": Mechanic#1
],
"input@[Reference)": Pipe#3,

```

```

        "output@[Reference)": Pipe#5,
        "hasStoredWater": false,
        "isBroken": false
    }
},
"1@[Reference)": Mechanic#4,
"2@[Reference)": Mechanic#9
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pipe#5,
    "2@[Reference)": Pipe#7,
    "3@[Reference)": Pump#2,
    "4@[Reference)": Pump#6,
    "5@[Reference)": Cistern#8
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Mechanic#1
},
"Timer#10":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#5,
        "2@[Reference)": Pipe#7,
        "3@[Reference)": Pump#2,
        "4@[Reference)": Pump#6,
        "5@[Reference)": Cistern#8
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#5,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#7
    ]
}
)
play
movePipe -p Pipe#7 -oe Pump#6 -ne Pump#2
endTurn
move -f Pump#2
changePumpDirection -ip Pipe#3 -op Pipe#7
endTurn
movePipe -p Pipe#7 -oe Pump#2 -ne Pump#6
    • Elvárt kimenet
"Game#0":
```

```
{
  "players": [
    {
      "0:Mechanic#1": {
        "activeField:Pump#2": {
          "neighbors": [
            {
              "0:Pipe#3": {
                "neighbors": [
                  {
                    "0@[Reference)": Pump#2
                  },
                  "players": [],
                  "oldWaterState": false,
                  "breakProofCounter": 0,
                  "slipperyCounter": 0,
                  "stickyCounter": 0,
                  "newWaterState": false,
                  "isBroken": false
                ],
                "1:Pipe#4": {
                  "neighbors": [
                    {
                      "0@[Reference)": Pump#2,
                      "1:Pump#5": {
                        "neighbors": [
                          {
                            "0@[Reference)": Pipe#4,
                            "1:Pipe#6": {
                              "neighbors": [
                                {
                                  "0@[Reference)": Pipe#6
                                }
                              ]
                            }
                          }
                        ]
                      }
                    }
                  ]
                }
              }
            }
          ]
        }
      }
    }
  ]
}

"0:Cistern#7": {
  "neighbors": [
    {
      "0@[Reference)": Pipe#6
    }
  ],
  "players": [
    {
      "0:Mechanic#8": {
        "neighbors": [
          {
            "0@[Reference)": Pipe#6
          }
        ]
      }
    }
  ]
}
```

```

    "activeField@[Reference)": Cistern#7
}

],
"input@[Reference)": Pipe#6
},

"1@[Reference)": Pump#5
], "players": [],
[], "oldWaterState": false,
false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
false,
"isBroken": false
}
],
"players": [],
[], "oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players": [
[ "0@[Reference)": Mechanic#1,
"1:Mechanic#9": {
{
"activeField@[Reference)": Pump#2
}
],
"input@[Reference)": Pipe#3,
"hasStoredWater": false,
"isBroken": false
}
],
"input@[Reference)": Pipe#4,
"hasStoredWater": false,
"isBroken": false
}
],
"players": []
}
},

```

```

    "1@[Reference)": Mechanic#9,
    "2@[Reference)": Mechanic#8
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pipe#4,
    "2@[Reference)": Pipe#6,
    "3@[Reference)": Pump#2,
    "4@[Reference)": Pump#5,
    "5@[Reference)": Cistern#7
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 2,
"activePlayer@[Reference)": Mechanic#8
},
"Timer#10":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pipe#4,
        "2@[Reference)": Pipe#6,
        "3@[Reference)": Pump#2,
        "4@[Reference)": Pump#5,
        "5@[Reference)": Cistern#7
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#4,
        "1@[Reference)": Pipe#3,
        "2@[Reference)": Pipe#6
    ]
}
}

```

## 8.2.28 Csúszós csövön csúszás kikapcsolt randommal

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassunk egy mozgató parancsot, a játékos rálép a célként megadott csőre és veszít egy akciót pontot. Mivel a célként választott cső csúszós, a játékos a cső egy általa választott szomszédjára kerül, mert a random opción kikapcsolt állapotban van. A mozgás sikeres, mert az adott csövön nem állt senki, a szomszédai pedig nem lehetnek csövek, tehát nem fordulhat elő, hogy nem tud rájuk lépni (csúszni) a játékos.

- Ellenőrzött funkcionalitás, várható hibahelyek

Csúszós cső működése kikapcsolt random mellett.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a játékos a csúszós csövön nem csúszik meg
4. a játék nem kérdezi meg a felhasználót a csúszás helyéről, pedig kikapcsolt random mellett ezt kéne tennie

- Bemenet

output

```

load -f tests/map/map28.txt
(
"Game#0":
{
  "players":
  [
    "0:Mechanic#1":
    {
      "activeField:Pump#2":
      {
        "neighbors":
        [
          "0:Pipe#3":
          {
            "neighbors":
            [
              "0@[Reference)": Pump#2,
              "1:Cistern#4":
              {
                "neighbors":
                [
                  "0@[Reference)": Pipe#3
                ],
                "players":
                [
                  "0:Saboteur#5":
                  {
                    "activeField@[Reference)": Cistern#4
                    }
                  ],
                  "input@[Reference)": Pipe#3
                }
              ],
              "players":
              [],
              "oldWaterState": false,
              "breakProofCounter": 0,
              "slipperyCounter": 0,
              "stickyCounter": 0,
              "newWaterState": false,
              "isBroken": false
            }
          ],
          "players":
          [
            "0@[Reference)": Mechanic#1
          ],
          "input@[Reference)": Pipe#3,
          "hasStoredWater": false,
        ]
      }
    }
  ]
}

```

```

                "isBroken": false
            }
        },
        "1@[Reference)": Saboteur#5
    ],
    "fields":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2,
        "2@[Reference)": Cistern#4
    ],
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2,
        "2@[Reference)": Cistern#4
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
)
play
move -f Pipe#1
makeSlippery
move -f Cistern#4
endTurn
move -f Pipe#3
random
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Cistern#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {

```

```

        "neighbors":
        [
            "0:Pump#4":
            {
                "neighbors":
                [
                    "0@[Reference)": Pipe#3
                ],
                "players":
                [
                    "0:Saboteur#5":
                    {
                        "activeField@[Reference)": Pump#4
                        }
                    ],
                    "input@[Reference)": Pipe#3,
                    "hasStoredWater": false,
                    "isBroken": false
                },
                "1@[Reference)": Cistern#2
            ],
            "players":
            [],
            "oldWaterState": false,
            "breakProofCounter": 0,
            "slipperyCounter": 4,
            "stickyCounter": 0,
            "newWaterState": false,
            "isBroken": false
        }
    ],
    "players":
    [
        "0@[Reference)": Mechanic#1
    ],
    "input@[Reference)": Pipe#3
}
},
"1@[Reference)": Saboteur#5
],
"fields":
[
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pump#4,
    "2@[Reference)": Cistern#2
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 2,

```

```

    "activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#4,
        "2@[Reference)": Cistern#2
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}

```

## 8.2.29 (összetett) Szabotőr csövet lyukaszt, ellép, szerelő odalép és megjavítja

- **Leírás**

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsokat: csövet lyukaszt, lépés, kört befejez, lépés, javítás. Az első három parancsot (csövet lyukaszt, lépés, kört befejez) az első, szabotőr játékos hajtja végre, melyek következtében rálép egy cső mezőre, majd kilyukasztja azt, végül átadja a kört a következő játékosnak. A következő két parancsot (lépés, javítás) a második, szerelő játékos hajtja végre, melyek következtében rálép az első játékos által kilyukasztott csőre és megjavítja azt. (A tesztelés során feltesszük, hogy a játékosoknak volt elég akciópontjuk a műveletek elvégzésére, valamint semmilyen egyéb szabályba sem ütköztek.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Aktív játékos lép, csövet lyukaszt, kört befejez, szerelő csövet javít.

Várható hibahelyek:

1. load-ban megadott fájl nem létezik
2. move-ban megadott mező nem létezik
3. a szabotőr játékos nem tudja kilyukasztani a csövet, amin áll, pedig minden követelménynek eleget tesz
4. a szerelő játékos nem tudja megjavítani a csövet, amin áll, pedig kéne tudnia

- **Bemenet**

```

output
load -f tests/map/map29.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":

```

```
[
    "0:Pipe#3":
    {
        "neighbors":
        [
            "0@[Reference)": Pump#2,
            "1:Cistern#4":
            {
                "neighbors":
                [
                    "0@[Reference)": Pipe#3
                ],
                "players":
                [
                    "0:Saboteur#5":
                    {
                        "activeField@[Reference)": Cistern#4
                        }
                    ],
                    "input@[Reference)": Pipe#3
                    }
                ],
                "players":
                [],
                "oldWaterState": false,
                "breakProofCounter": 0,
                "slipperyCounter": 0,
                "stickyCounter": 0,
                "newWaterState": false,
                "isBroken": false
                }
            ],
            "players":
            [
                "0@[Reference)": Mechanic#1
            ],
            "input@[Reference)": Pipe#3,
            "hasStoredWater": false,
            "isBroken": false
            }
        },
        "1@[Reference)": Saboteur#5
    ],
    "fields":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2,
        "2@[Reference)": Cistern#4
    ],
]
```

```

    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 3,
    "activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2,
        "2@[Reference)": Cistern#4
    ],
    "statefuls":
    [
        "0@[Reference)": Pipe#3
    ]
}
)
play
move -f Pipe#3
breakField //Pipe#3
move -f Pump#2
endTurn
move -f Pipe#3
repairField
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [
                    "0:Pump#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pipe#2
                        ],
                        "players":
                        [
                            "0:Saboteur#4":
                            {
                                "activeField@[Reference)": Pump#3
                            }
                        ]
                    }
                ]
            }
        }
    ]
}

```

```

        ],
        "input@[Reference)": Pipe#2,
        "hasStoredWater": false,
        "isBroken": false
    },
    "1:Cistern#5":
    {
        "neighbors":
        [
            "0@[Reference)": Pipe#2
        ],
        "players":
        [],
        "input@[Reference)": Pipe#2
    }
],
"players":
[
    "0@[Reference)": Mechanic#1
],
"oldWaterState": false,
"breakProofCounter": 3,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
},
"1@[Reference)": Saboteur#4
],
"fields":
[
    "0@[Reference)": Pipe#2,
    "1@[Reference)": Pump#3,
    "2@[Reference)": Cistern#5
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 1,
"activePlayer@[Reference)": Saboteur#4
},
"Timer#6":
{
    "periodics":
    [
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Pump#3,
        "2@[Reference)": Cistern#5
    ],
    "statefuls":

```

```

    [
        "0@[Reference)": Pipe#2
    ]
}

```

## 8.2.30 (összetett) Szerelő megjavít egy elromlott pumpát, befejezi a körét, egy másik odalép és átállítja a pumpát

- Leírás

A bemeneti fájlból vagy a szabványos bemenetről beolvassuk a következő parancsokat: javítás, kört befejez, lépés, pumpát átállít. Az első két parancsot ( javítás, kört befejez) az első, szerelő játékos hajtja végre, melyek következtében megjavítja a pumpát, amin áll, majd befejezi a körét. A következő két parancsot (lépés, pumpát átállít) a második, szintén szerelő játékos hajtja végre, melyek következtében rálép az első játékos által megjavított pumpára és átállítja rajta a víz folyási irányát. (A tesztelés során feltesszük, hogy a játékosoknak volt elég akciót pontjuk a műveletek elvégzésére, valamint semmilyen egyéb szabályba sem ütköztek.)

- Ellenőrzött funkcionalitás, várható hibahelyek

Aktív szerelő játékos pumpát javít, kört befejez, lépés, pumpát átállít.

Várható hibahelyek:

- load-ban megadott fájl nem létezik
- move-ban megadott mező nem létezik
- a második játékos nem tud rálépn a pumpára az első játékos ottléte miatt, pedig egyszerre több embernek is kellene tudnia egy pumpán tartózkodni
- az első játékos nem tudja megjavítani a pumpát, amin áll, pedig kéne tudnia
- a changePumpDirection parancs paraméterei érvénytelenek vagy szabályellenesek
- a második játékos nem tudja átállítani a pumpát, pedig kéne tudnia

- Bemenet

```

output
load -f tests/map/map30.txt
(
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pump#2":
            {
                "neighbors":
                [
                    "0:Pipe#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pump#2,
                            "1:Cistern#4":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#3
                                ],
                            }
                        ],
                    }
                ],
            }
        }
    ]
}

```

```

    "players": [
        {
            "0:@[Reference)": Saboteur#5:
        }
    ],
    "input@[Reference)": Pipe#3
},
],
"players": [],
"oldWaterState": false,
"breakProofCounter": 0,
"slipperyCounter": 0,
"stickyCounter": 0,
"newWaterState": false,
"isBroken": false
}
],
"players": [
    {
        "0@[Reference)": Mechanic#1
    },
    "input@[Reference)": Pipe#3,
    "hasStoredWater": false,
    "isBroken": false
}
},
"1@[Reference)": Saboteur#5
],
"fields": [
    "0@[Reference)": Pipe#3,
    "1@[Reference)": Pump#2,
    "2@[Reference)": Cistern#4
],
"saboteurPoints": 0,
"mechanicPoints": 0,
"actionNumber": 3,
"activePlayer@[Reference)": Saboteur#5
},
"Timer#6":
{
    "periodics": [
        "0@[Reference)": Pipe#3,
        "1@[Reference)": Pump#2,
        "2@[Reference)": Cistern#4
    ]
}

```

```

        ],
        "statefuls":
        [
            "0@[Reference)": Pipe#3
        ]
    }
)
play
repaireField //Pump#2
endTurn
move -f Pump#2
changePumpDirection -ip Pipe#3 -op Pipe#3
    • Elvárt kimenet
"Game#0":
{
    "players":
    [
        "0:Mechanic#1":
        {
            "activeField:Pipe#2":
            {
                "neighbors":
                [
                    "0:Pump#3":
                    {
                        "neighbors":
                        [
                            "0@[Reference)": Pipe#2
                        ],
                        "players":
                        [
                            "0:Saboteur#4":
                            {
                                "activeField@[Reference]":
Pump#3
                                    }
                                ],
                                "input@[Reference)": Pipe#2,
                                "hasStoredWater": false,
                                "isBroken": false
                            },
                            "1:Cistern#5":
                            {
                                "neighbors":
                                [
                                    "0@[Reference)": Pipe#2
                                ],
                                "players":
                                [],
                                "input@[Reference)": Pipe#2
                            }
                        ]
                    }
                ]
            }
        ]
    ]
}

```

```

        }
    ],
    "players":
    [
        {
            "0@[Reference)": Mechanic#1
        },
        "oldWaterState": false,
        "breakProofCounter": 3,
        "slipperyCounter": 0,
        "stickyCounter": 0,
        "newWaterState": false,
        "isBroken": false
    }
},
"1@[Reference)": Saboteur#4
],
"fields":
[
    {
        "0@[Reference)": Pipe#2,
        "1@[Reference)": Pump#3,
        "2@[Reference)": Cistern#5
    },
    "saboteurPoints": 0,
    "mechanicPoints": 0,
    "actionNumber": 1,
    "activePlayer@[Reference)": Saboteur#4
},
"Timer#6":
{
    "periodics":
    [
        {
            "0@[Reference)": Pipe#2,
            "1@[Reference)": Pump#3,
            "2@[Reference)": Cistern#5
        },
        "statefuls":
        [
            {
                "0@[Reference)": Pipe#2
            }
        ]
}
}

```

### 8.3 A tesztelést támogató programok tervei

A prototípus kimenetét manuálisan kell összehasonlítani a tesztesetnél specifikált kimenettel. Ebben az esetben a kiértékelés biztosan teljes, azonban érdemes a kimenet szemantikus jelentésére is figyelni, ha például eltörünk egy csövet, akkor a parancs kiadása után a kimenetben az adott pumpának az **isBroken** értéke **true** lesz. Ha átkötünk egy csövet, az másik pumpához lesz csatlakoztatva. A prototípus kimenete időnként nehezen átlátható, azonban minden fontos információt tartalmaz, a segítségével tökéletesen és hiánytalanul meg lehetne jeleníteni a pálya állapotát.

A dokumentumban specifikált, egyes tesztesetekhez tartozó elvárt kimeneteket az átláthatóság érdekében a prototípus programmal együtt külön-külön fájlokban fogjuk átadni, ezeket a fájlokat érdemes olyan szövegszerkesztővel megjeleníteni (pl. notepad), ahol lehet állítani a betűméretet, a fontos ugyanis az, hogy vízszintesen legyen elég hely az egymásba ágyazott JSON objektumoknak.

**Napló**

Kezdet	Időtartam	Résztvevők	Leírás
2023.04.27. 21:20	1 óra	Zavadil	Fedlap elkészítése + merge, követelmények kiegészítése
2023.04.27. 12:00	1,5 óra	Völgyesi Zavadil Garai	Értekezlet. Eredmény: Völgyesi, Garai, Varga dolgozza ki a teszteseteket, Mali és Zavadil az osztályok leírását. Zavadil csinál Java -> JSON parser-t és elkészíti a 8.0 fejezetet.
2023.04.28. 10:00	6 óra	Zavadil	Java -> JSON parser megírása az elvárt kimenetek könnyebb létrehozásához
2023.04.29. 20:00	2 óra	Völgyesi	Teszt tervezés írása
2023.04.29. 21:00	1,5 óra	Garai	Teszt kimenetek írása, várható hibahelyek írása
2023.04.30. 9:00	1 óra	Völgyesi	Teszt tervezés kiegészítése
2023.04.30. 11:30	2 óra	Zavadil	8.0 megírása
2023.04.30 14:00	7 óra	Zavadil	8.1.1-8.1.6 megírása, parser javítása, módosítások (8.0) kiegészítése
2023.04.30 20:00	3 óra	Varga	8.2.15-8.2.18 tesztesetek megírása,
2023.05.01. 12:00	1,5 óra	Völgyesi	További teszt tervezés írása
2023.05.01 18:00	2 óra	Varga	8.2.28-8.2.30 tesztesetek megírása
2023.04.30 16:00	5 óra	Garai	Tesztek kimeneteinek írása 1-13 Várható hibahelyek írása
2023.05.01 17:00	4 óra	Garai	Tesztek kimeneteinek írása 22-27 Várható hibahelyek írása
2023.05.01. 22:00	2 óra	Mali	8.1.7-8.1.10 megírása
2023.05.02. 16:00	1,5 óra	Mali	8.1.11-8.1.12 megírása
2023.05.02 19:00	0,5 óra	Garai	Tesztek bemeneteinél a load parancs fájljainak sorszámozása, ahol láttam hashtagek javítása

## 10. Prototípus beadása

### 10.1 Fordítási és futtatási útmutató

#### 10.1.1 Fájllista

Fájl neve	Méret (byte)	Keletkezés ideje	Tartalom
Cistern.java	2989	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JavaToJSONParser.java	4156	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JSONEntity.java	2343	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JSONList.java	981	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JSONObject.java	1868	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JSONPrimitive.java	616	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JSONReference.java	480	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
C_JSON_Serializable.java	397	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Field.java	3741	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Game.java	7092	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Main.java	38003	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Mechanic.java	3817	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Periodic.java	389	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Pipe.java	8084	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Player.java	2696	2023-05-14	Forrásfájl a vele azonos nevű osztályhoz

		20:59:51	
Prototype.java	16814	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Pump.java	4404	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Saboteur.java	940	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Source.java	1121	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Stateful.java	264	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
Timer.java	3005	2023-05-14 20:59:51	Forrásfájl a vele azonos nevű osztályhoz
expected_output1	832	2023-05-13 11:43:47	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output10	1255	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output11	1255	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output12	1255	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output13	1255	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output14	1744	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output15	1225	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output16	551	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output17	695	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output18	1317	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output19	1242	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output2	1126	2023-05-14	A fájlnév végén lévő számjegy(ek)hez tartozó

		12:19:09	teszt elvárt kimenete
expected_output20	1716	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output21	1716	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output22	2047	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output23	871	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output24	1744	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output25	2675	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output26	2969	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output27	2521	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output28	1257	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output29	1210	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output3	953	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output30	1210	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output4	591	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output5	592	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output6	592	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output7	592	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output8	592	2023-05-14 12:19:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt elvárt kimenete
expected_output9	1094	2023-05-14	A fájlnév végén lévő számjegy(ek)hez tartozó

		12:19:09	teszt elvárt kimenete
input10.txt	44	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input11.txt	44	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input12.txt	44	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input13.txt	44	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input14.txt	42	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input15.txt	42	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input16.txt	43	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input17.txt	43	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input18.txt	77	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input19.txt	81	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input1.txt	53	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input20.txt	78	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input21.txt	78	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input22.txt	93	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input23.txt	75	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input24.txt	64	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input25.txt	162	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input26.txt	146	2023-05-14	A fájlnév végén lévő számjegy(ek)hez tartozó

		20:43:36	teszt bemeneti parancsait tartalmazza
input27.txt	191	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input28.txt	102	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input29.txt	112	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input2.txt	50	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input30.txt	128	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input3.txt	50	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input4.txt	46	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input5.txt	46	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input6.txt	47	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input7.txt	47	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input8.txt	48	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
input9.txt	46	2023-05-14 12:26:13	A fájlnév végén lévő számjegy(ek)hez tartozó teszt bemeneti parancsait tartalmazza
map10.txt	1014	2023-05-14 13:00:03	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map11.txt	1014	2023-05-14 13:00:10	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map12.txt	1014	2023-05-14 13:00:16	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map13.txt	1014	2023-05-14 13:00:23	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map14.txt	1018	2023-05-14 12:50:42	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map15.txt	975	2023-05-14	A fájlnév végén lévő számjegy(ek)hez tartozó

		20:43:36	teszt pályaképét tartalmazza
map16.txt	760	2023-05-14 12:51:06	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map17.txt	803	2023-05-14 12:51:20	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map18.txt	972	2023-05-14 12:51:32	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map19.txt	1056	2023-05-14 12:51:41	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map1.txt	838	2023-05-14 12:42:55	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map20.txt	1014	2023-05-14 12:51:51	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map21.txt	1014	2023-05-14 12:52:03	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map22.txt	1081	2023-05-14 12:52:14	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map23.txt	836	2023-05-14 12:52:29	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map24.txt	1018	2023-05-14 12:52:39	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map25.txt	1211	2023-05-14 12:52:49	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map26.txt	1259	2023-05-14 12:52:59	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map27.txt	1230	2023-05-14 12:53:09	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map28.txt	1014	2023-05-14 12:53:17	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map29.txt	1014	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map2.txt	975	2023-05-14 12:43:20	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map30.txt	1067	2023-05-14 20:43:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map3.txt	877	2023-05-14	A fájlnév végén lévő számjegy(ek)hez tartozó

		12:46:01	teszt pályaképét tartalmazza
map4.txt	695	2023-05-14 12:46:25	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map5.txt	695	2023-05-14 12:46:36	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map6.txt	719	2023-05-14 12:46:47	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map7.txt	719	2023-05-14 12:46:58	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map8.txt	695	2023-05-14 12:47:10	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
map9.txt	955	2023-05-14 12:47:21	A fájlnév végén lévő számjegy(ek)hez tartozó teszt pályaképét tartalmazza
mapDefault.txt	838	2023-05-14 12:59:48	Az alapértelmezett pályaképet tartalmazza

### 10.1.2 Fordítás

A 10.1.1 listában leírt összes “java” kiterjesztésű fájlt (és egyelőre csak ezeket) helyezzük bele egy üres mappába. A többi fájlt a következőképp helyezzük el: először is, hozunk létre a “java” kiterjesztésű fájlok mappájában egy almappát, “tests” néven. Ezután:

- az “input” kezdetű fájlokat helyezzük el egy “input” nevű mappában a “tests” mappán belül.
- a “map” kezdetű fájlokat helyezzük el egy “map” nevű mappában a “tests” mappán belül.

A program futtatásához és fordításához ezzel minden szükséges fájlt elhelyeztünk. Az “expected\_output” kezdetű fájlok a program futtatásához ugyan nem szükségesek, azonban a kimenetek manuális összehasonlítását hivatottak segíteni.

Ezután ellenőrizzük, hogy a java és javac parancsok fel vannak-e telepítve, majd adjuk ki a következő parancsot a fordításhoz:

```
javac *.java
```

### 10.1.3 Futtatás

A 10.1.2. fejezetben leírt lépések után ugyanabban a mappában adjuk ki a következő parancsot a futtatáshoz:

```
java Main
```

Ezt követően a program elindul, megjelenik a parancsértelemező interfész, először egy listával a lehetséges parancsokról. Itt minden parancs a “Prototípus elkészítése” című dokumentum szerint működik.

*Amennyiben követtük a fordítási útmutatót, az egyes tesztek pályaképeihez a “tests/map/mapX.txt” elérési úttal, valamint a tesztekhez szükséges kiadandó parancsok fájljaihoz a “tests/input/inputX.txt” elérési úttal juthatunk, ahol X jelképezi az X-edik tesztet.*

Például a “Prototípus elkészítése” című dokumentumban definiált 5. teszt a következő parancs kiadásával indítható:

```
input -f tests/input/input5.txt
```

## 10.2 Tesztek jegyzőkönyvei

### 10.2.1 Teszteset1

Teszteleő neve	Garai Donát
Teszt időpontja	2023.05.14.

### 10.2.2 Teszteset2

Teszteleő neve	Garai Donát
Teszt időpontja	2023.05.14.

### 10.2.3 Teszteset3

Teszteleő neve	Garai Donát
Teszt időpontja	2023.05.14.

### 10.2.4 Teszteset4

Teszteleő neve	Garai Donát
Teszt időpontja	2023.05.14.

### 10.2.5 Teszteset5

Teszteleő neve	Garai Donát
Teszt időpontja	2023.05.14.

**10.2.6 Teszteset6**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

**10.2.7 Teszteset7**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

**10.2.8 Teszteset8**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

**10.2.9 Teszteset9**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

**10.2.10 Teszteset10**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	A víz nem folyik.
<b>Lehetséges hibaok</b>	Az endTurn parancs nem működik, mert a Timer nincsen kiszerelizálva fájlba.
<b>Változtatások</b>	A serialize függvény a Timer-t is kimenti fájlba, így fel lesz töltve load parancs utáni beolvasáskor.

**10.2.11 Teszteset11**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

**10.2.12 Teszteset12**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	A szerelők eggyel több pontot kapnak mint kéne.
<b>Lehetséges hibaok</b>	A Cistern kétszer kéri le a vizet az input csőtől.
<b>Változtatások</b>	A Cistern step() metódusának javítása.

**10.2.13 Teszteset13**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

**10.2.14 Teszteset14**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

**10.2.15 Teszteset15**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

**10.2.16 Teszteset16**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

**10.2.17 Teszteset17**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

**10.2.18 Teszteset18**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	Hibát dob a movePipe parancs során.
<b>Lehetséges hibaok</b>	A parancsértelmező rossz sorrendet feltételez.
<b>Változtatások</b>	A második és harmadik paraméter típusa Pump.

**10.2.19 Teszteset19**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	Megváltozik a cső vége.
<b>Lehetséges hibaok</b>	A paraméterként átadott függvény megfelel a kritériumoknak.
<b>Változtatások</b>	movePipe -p Pipe#3 -oe Source#5 -ne Source#5 parancsot kell használni.

**10.2.20 Teszteset20**

Tesztelő neve	Varga Dávid
Teszt időpontja	2023.05.14.

**10.2.21 Teszteset21**

Tesztelő neve	Varga Dávid
Teszt időpontja	2023.05.14.

Tesztelő neve	Varga Dávid
Teszt időpontja	2023.05.14.
Teszt eredménye	Átáll a ki és bemenet.
Lehetséges hibaok	A parancsnak rosszak a paraméterei.
Változtatások	“changePumpDirection -ip Pipe#3 -op Pipe#4” a helyes paraméterezés

**10.2.22 Teszteset22**

Tesztelő neve	Garai Donát
Teszt időpontja	2023.05.14.

**10.2.23 Teszteset23**

Tesztelő neve	Garai Donát
Teszt időpontja	2023.05.14.

**10.2.24 Teszteset24**

Tesztelő neve	Garai Donát
Teszt időpontja	2023.05.14.

**10.2.25 Teszteset25**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	Nem jók a bemeneti parancsok, melyeknél hashtagelés van.
<b>Lehetséges hibaok</b>	Mivel minden parancs után kiírja a kimenetet, ezért a hashtags folyamatosan változnak.
<b>Változtatások</b>	Bemeneti fájlban a hashtags javítása.

**10.2.26 Teszteset26**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	A szabotőr pumpán kezd cső helyett.
<b>Lehetséges hibaok</b>	Az eredeti tesztleírás rossz, ezáltal az eredeti elvárt kimenet is rossz. Az eredeti tesztleírásban helytelenül van írva, hogy a szabotőr csövön kezd és azt lyukasztja. A helyes leírás, hogy a szabotőr pumpán kezd, lyukasztani próbálja, de nem sikerül neki, mivel az nem cső.
<b>Változtatások</b>	Az elvárt kimenet átírása.

**10.2.27 Teszteset27**

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Garai Donát
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	A ciszternán spawnolt pumpa.
<b>Lehetséges hibaok</b>	A ciszterna step metódusában nincs ellenőrizve, hogy a random milyen értéket vesz fel.
<b>Változtatások</b>	Ciszterna step függvényének javítása.

**10.2.28 Teszteset28**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	A Saboteur és Mechanic helyzete fel van cserélve .
<b>Lehetséges hibaok</b>	A amikor a szabotőr lép akkor rossz helyre lép a csőről.
<b>Változtatások</b>	move -f Cistern helyet move -f Pump#4

**10.2.29 Teszteset29**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

**10.2.30 Teszteset30**

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.

<b>Tesztelő neve</b>	Varga Dávid
<b>Teszt időpontja</b>	2023.05.14.
<b>Teszt eredménye</b>	A második játékos egy szabotőr. Lehetséges hogy a két játékos sorrendje nem lesz jó, viszont ugyanaz a logikája a kiírásnak.

<b>Lehetséges hibaok</b>	Rosszul volt megírva a teszeset.
<b>Változtatások</b>	Az új parancsok a következők: load -f tests/map/map30.txt play move -f Pipe#3 move -f Pump#3 repairField endTurn changePumpDirection -ip Pipe#3 -op Pipe#5 end

### 10.3 Értékelés

Tag neve	Tag neptun	Munka százalékban	Aláírás
Garai Donát	CRSL00	20	<i>Garai Donát</i>
Varga Dávid	Z8JXEQ	20	<i>VARGA DÁVID</i>
Mali Bence	ZA2ENI	20	
Zavadil Balázs Dávid	FOIFDF	20	
Völgyesi Soma	GBI4MD	20	

## 10.4 Napló

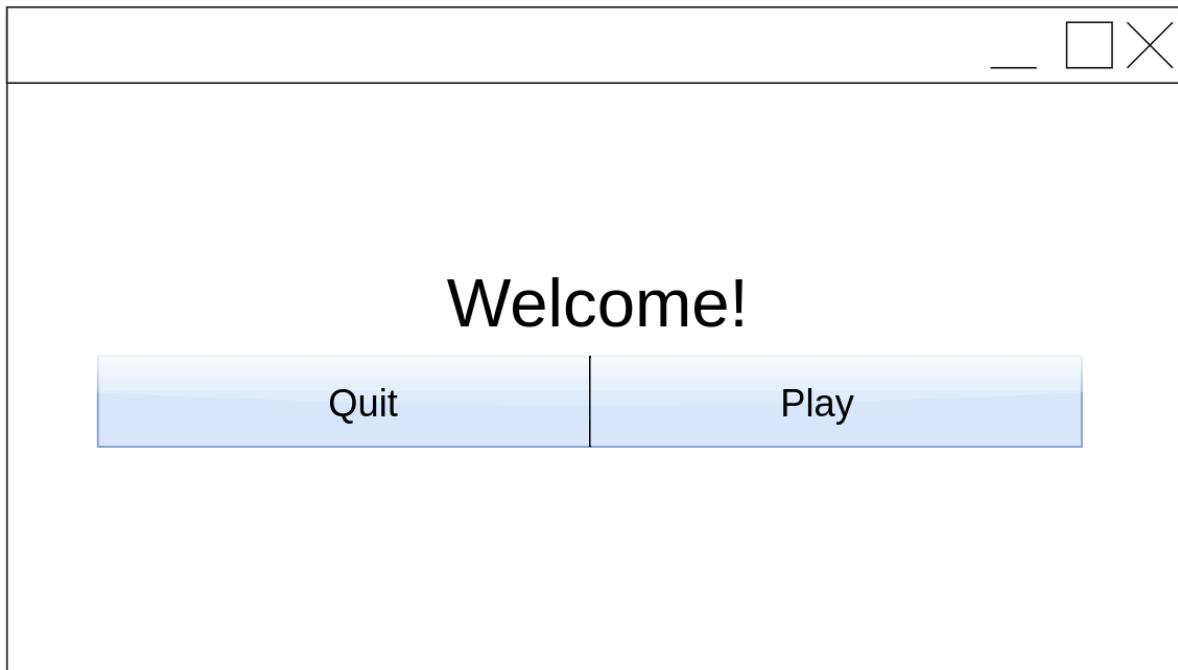
Kezdet	Időtartam	Résznevők	Leírás
2023.05.12. 14:00	1 óra	Mali Varga Völgyesi Garai Zavadil	Értekezlet. Feladatok kiosztása
2023.05.12. 17:00	4 óra	Mali	Metódusok befejezése, Javadoc kommentek megírása
2023.05.12. 18:00	3 óra	Zavadil	Prototípus interfész elkezdése
2023.05.13. 09:30	4 óra	Zavadil	Prototípus interfész befejezése, fájlba írás, fájlból olvasás, 1. teszt elkészítése (példának)
2023.05.13. 14:00	1.5 óra	Zavadil	JSON parser és prototípus javadoc kommentek, 10.1.2, 10.1.3 megírása
2023.05.13. 14:00	2 óra	Völgyesi	2.-30. tesztek elkészítése
2023.05.14. 12:00	2 óra	Varga	Tesztesetek ellenőrzés, kimeneti fájlok átírása, hogy helyesen működjön
2023.05.14. 16:00	2 óra	Varga	Tesztesetek átírása mert nem volt megfelelő.
2023.05.14. 16:00	1 óra	Zavadil	Prototípus interfész hibák javítása, 10.1.2, 10.1.3 bővítése, pályaképek újragenerálása
2023.05.14. 11:00	2 óra	Garai	Tesztek ellenőrzése, kód javítása, bemenetek javítása
2023.05.14. 14:00	3 óra	Garai	Tesztek ellenőrzése, kód javítása, bemenetek javítása
2023.05.14. 17:30	0,5 óra	Mali	Javítások a prototípus osztályaiban
2023.05.14. 20:30	2 óra	Zavadil	Fájllista létrehozása, helyesírási hibák javítása, tesztelés, fájlok kigyűjtése, online beadás, nyomtatás

## 11. Grafikus felület specifikációja

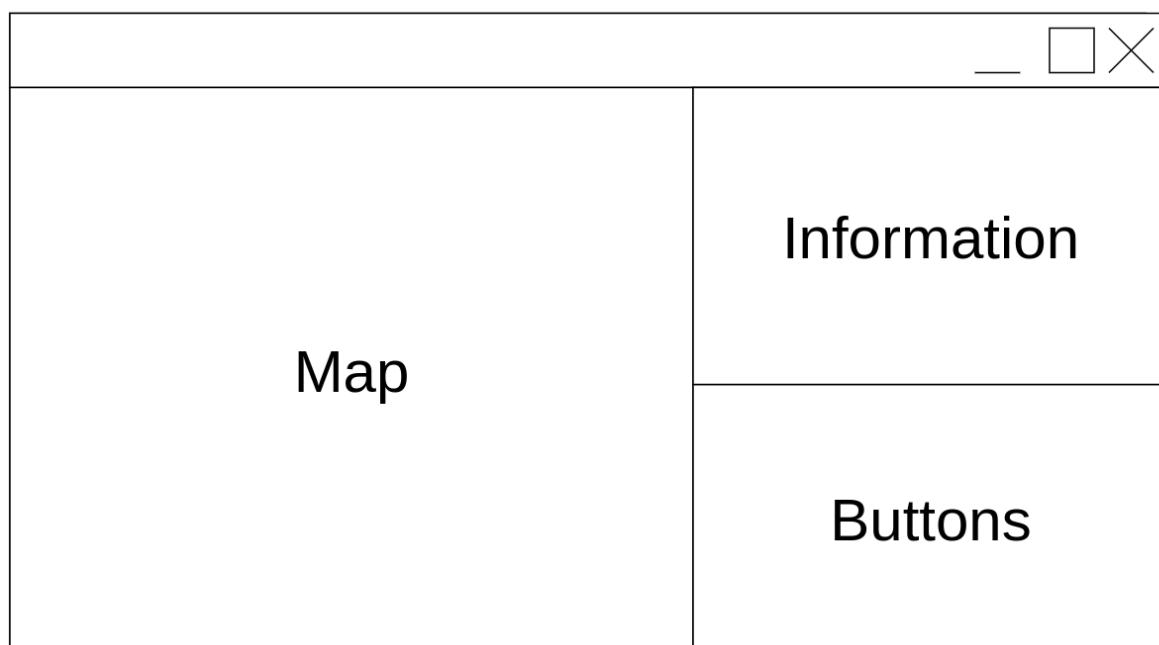
### 11.1 A grafikus interfész

A program elindítása után az üdvözlő szöveg ("Welcome!") mellett egy menü jelenik meg az alkalmazás megnyitása után, melyen két gomb szerepel:

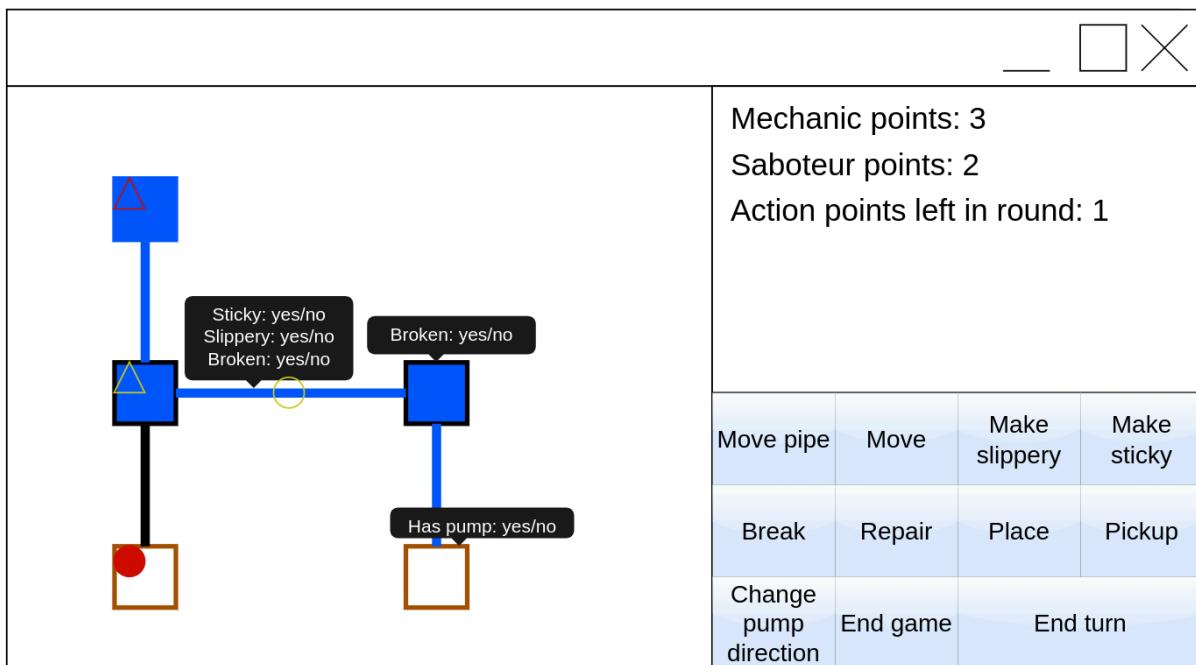
- "Play" gomb: elindul a játék ennek hatására és megjelenik a pálya
- "Quit" gomb: kilép a játékból



Ezután a játék felületéről egy absztrakt kép:



A játék egy lehetséges állapotáról egy részletes kép:



*Megjegyzés: a fenti képen egyszerre 3 tooltip látszik, ez nem lehetséges, hiszen ezek akkor jelennének csak meg, ha a hozzájuk tartozó objektum fölött visszük az egeret, azonban jól szemlélteti az információk hollétét.*

A képernyő bal oldalán a pálya, a jobb oldalának felső részén szöveges elemek találhatók, melyek a következő információkat írják ki:

- Szabotörök és szerelők pontjai
- Körben hátralévő akciók száma

A képernyő jobb oldalának alsó felén a következő gombok találhatók meg:

- **Move pipe:** Cső mozgatása, ezután háromszor kell kattintani:
  - Először egy csőre, ezzel kiválasztva melyik cső végét szeretnénk mozgatni
  - Utána ennek a csőnek az egyik végén lévő objektumra, ezzel kijelölve a cső végét, melyet mozgatni szeretnénk. Amennyiben egy csőnek a mozgatni kívánt vége nincs sehol csatlakoztatva, akkor automatikusan az választódik ki csatlakozásra, függetlenül a kattintás helyétől.
  - Végül egy olyan objektumot kell választani, amelyhez csatlakoztatni szeretnénk a cső végét. Amennyiben minden objektumon kívül, vagy hibás objektumra (pl. másik cső vagy ugyanaz az objektum, ahol a cső másik vége van) kattintunk, a cső kiválasztott vége sehol nem lesz kötve, "lógni fog".
- **Move:** Az aktív játékos mozgatása, gombnyomás után választani kell egy célpont mezőt (a pályán kattintással).
- **Make slippery:** Az aktív játékos mezőjét csúszóssé teszi, amennyiben ez lehetséges.
- **Make sticky:** Az aktív játékos mezőjét ragadóssé teszi, amennyiben ez lehetséges.
- **Break:** Az aktív játékos mezőjét eltöri, amennyiben ez lehetséges.
- **Repair:** Az aktív játékos mezőjét megjavítja, amennyiben ez lehetséges.
- **Place:** Az aktív játékos mezőjére lerakat vele egy pumpát, amennyiben ez lehetséges.

- **Place:** Az aktív játékos mezőjéről felvetet vele egy pumpát, amennyiben ez lehetséges.
- **Change pump direction:** Megváltoztatja a pumpa bemeneti (gombnyomás után első kattintás) és kimeneti (gombnyomás után második kattintás) csöveit.
- **End game:** Befejezi a játékot, az nyer akinek jelenleg több pontja van.
- **End turn:** Befejezi az aktív játékos körét.

### Játékosok a pályán:

- **Szerelő (Mechanic):** piros és sárga háromszög (2 szerelő van a játékban).
- **Szabotőr (Saboteur):** piros és sárga kör (2 szabotőr van a játékban).
- ha ő az aktív játékos akkor ki van töltve pirossal illetve sárgával



### Cső (Pipe):

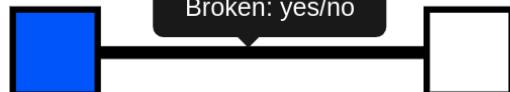
- *nincs benne víz: fekete egyenes vonal két másik elem között*
- *van benne víz: kék egyenes vonal két másik elem között*
- *ragadós, csúszós, törött állapot jelzése: megjelenik egy tooltip amennyiben rávisszük a kurzort, erről leolvashatók ezek az adatok*
- *van rajta játékos: a közepén jelenik meg az adott játékos ikonja (egyszerre csak 1 játékos tartózkodhat a csövön a játékszabályok értelmében)*

Cső, melyben folyik víz, rajta egy játékossal:



Sticky: yes/no  
Slippery: yes/no  
Broken: yes/no

Cső, melyben nem folyik víz, rajta 0 játékossal:



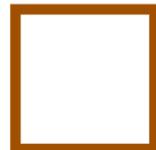
Az egeret felé vittük, ezért megjelentek az adatok a tooltip-ben

### Ciszterna (Cistern):

- barna keretes négyzet

- rajta a játékosok 4 lehetséges helyen lehetnek (bal felül, bal alul, jobb felül, jobb alul), attól függően lesznek itt elhelyezve, hogy hányan állnak már rajta, amikor rálépnek
- van-e rajta pumpa: megjelenik egy tooltip amennyiben rávisszük a kurzort, ebből olvasható ki

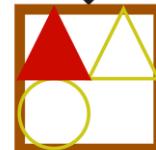
0 játékossal



Ciszterna:

3 játékossal (egér felette)

Has pump: yes/no

**Forrás (Source):**

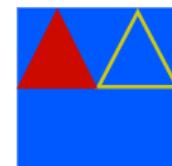
- keret nélküli négyzet, minden kék a belseje
- rajta a játékosok 4 lehetséges helyen lehetnek (bal felül, bal alul, jobb felül, jobb alul), attól függően lesznek itt elhelyezve, hogy hányan állnak már rajta, amikor rálépnek

0 játékossal

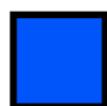


Forrás:

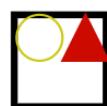
2 játékossal

**Pumpa (Pump):**

- fekete keretes négyzet, ha van a tartályában víz kék kitöltéssel, ha nincs, fehér kitöltéssel
- rajta a játékosok 4 lehetséges helyen lehetnek (bal felül, bal alul, jobb felül, jobb alul), attól függően lesznek itt elhelyezve, hogy hányan állnak már rajta, amikor rálépnek
- törött: megjelenik egy tooltip amennyiben rávisszük a kurzort, ebből olvasható ki.

0 játékossal  
víz van a tartályában

Pumpa:

2 játékossal  
víz nincs a tartályában2 játékossal  
víz nincs a tartályában  
egér felette

Broken: yes/no



A játék elindításakor egy fix beépített pálya jelenik meg, amelyet biztosan átláthatóan rajzol ki a program. Ezt követően a játékosok ezt az esztétikus képet tönkretehetik (pl. lerakhatnak sok pumpát), ezzel átláthatatlanná téve a játékeret. Ennek elkerüléséért, középső egérgomb segítségével a pumpák, ciszternák és források "megfoghatók" és el lehet őket mozgatni máshov.

## 11.2 A grafikus rendszer architektúrája

### 11.2.1 A felület működési elve

Az osztálydiagramban látszik, hogy a grafikus osztály csak egyetlen másik osztállyal van közvetlen kapcsolatban, a Game osztállyal. Ennek célja hogy könnyen lehessen módosítani, megteremtve tehát, annak lehetőségét is, hogy akár, egy másik grafikus osztállyal le lehessen váltani.

Ennek következménye hogy a Graphics osztály csak megjelenítéshez szükséges információt tárol és kezel, mik a játék logikáját a game osztály, és a többi objektum valósítja meg.

A megjelenítés és a logika közötti kommunikáció alapelve pull alapú: a Graphics osztály minden egyes gombnyomás esetén meghívja a megfelelő metódusát a Game osztályon, vagy annak egyik változóján, majd a legvégén a pull() metódussal kirajzolja újra a pályát, független attól hogy a művelet sikeres volt-e, vagy történt-e bármilyen változás.

A pull() metódus tehát a megjelenítés alapköve, azért felelős, hogy kérdezze a játéktól az új állapotát:

- *getFields()* metódussal a pálya csomópontjait frissíti
- *getPlayers()* ahhoz szükséges, hogy a pályán tudjuk melyik csomópontokon helyezkedne el a játékosok
- *getActivePlayer()*: az aktív játékost újra kell rajzolni telített formában
- *getActionNumber()*: az akciópontok frissítéséhez szükséges
- *getSaboteurPoints()*: a szabotőr pontjait frissíti
- *getMechanicPoints()*: a szerelő pontjait frissíti
- *getActionNumber()*: az aktív játékos pontszámának megjelenítéséhez kell.

Nem utolsó sorban a pull() metódus végén, ha változott az objektumok száma (le lett téve egy új pumpa, vagy a szerelők egy új csövet tettek le), akkor megfelelően létrehoz új objektumot is ami ezt képviseli, majd minden ilyen objektumra meghívja a draw() függvényt.

## **11.2.2A felület osztály-struktúrája**



## 11.3 A grafikus objektumok felsorolása

### 11.3.1 Game

- **Új Attribútumok**
  - **-List<Pump> pumps:** Az összes pumpát számontartó lista
  - **-List<Cistern> cisterns:** Az összes ciszternát számontartó lista
  - **-List<Source> sources:** Az összes forrást számontartó lista
  - **-List<Saboteur> saboteurs:** *A játékban szereplő szabotőröket tárolja*
  - **-List<Mechanic> mechanics:** *A játékban szereplő szerelőket tárolja*
  - **-List<Pipe> pipes:** Az összes csövet számontartó lista
- **Megváltozott/Új Metódusok**
  - **+bool mechanicWin():** Visszaadja, hogy a szerelő csapat elérte-e már a kellő pontszámot a győzelemhez
  - **+bool saboteurWin():** Visszaadja, hogy a szabotőr csapat elérte-e már a kellő pontszámot a győzelemhez

### 11.3.2 DisplayCistern

- **Felelősség**  
Egy ciszternát reprezentál a grafikus megjelenítés során, referenciát tárol a hozzá tartozó ciszternához.
- **Ősosztályok**  
DisplayNode
- **Interfészek**  
-
- **Attribútumok**
  - **-Cistern c:** referencia a Cistern objektumra, amelyet a CisternNode objektum reprezentál a grafikus megjelenítésnél
  - **-Color color:** *A kirajzolt ciszterna megjelenítési és körvonal színe.*
    - **+void draw():** Kirajzolja a bennfoglaló téglalapját **color** kitöltő és körvonal színnel.
    - **+void getGameReference():** Visszaadja a reprezentált ciszterna referenciáját (*c*)
    - **+String getTooltipText():** Visszaadja a következő sorokat tartalmazó String-et:
      - “*Cistern*”
      - “*Has pump: <yes/no>*”: *c.hasPump()* függvényében
- **Metódusok**
  - **+void draw():** Kirajzolja a bennfoglaló téglalapját **color** kitöltő és körvonal színnel.
  - **+void getGameReference(): Cistern:** A metódus visszaadja **c-t**.

- `+String getToolTipText():` Visszaadja a következő sorokat tartalmazó Stringet:  
“Cistern”  
“Has pump: <yes/no>”

### 11.3.3 DisplayField

- **Felelősség**

Absztrakt ōsosztály a megjelenítendő Field leszármazott elemek megjelenített objektumaihoz.

- **Ósosztályok**
- 

- **Interfészek**
- 

- **Attribútumok**
- `#List<PlayerIcon> playerIcons:` A megjelenített mezőn tartózkodó játékosok ikonjai.

- **Metódusok**
- `#abstract void setPlayerIcons():` Beállítja a rajta tartózkodó játékosok ikonjait (`playerIcons`).
- `+abstract Field getGameReference():` Visszaadja azt a **Field** típusú objektumot, melyet reprezentál.
- `+public abstract void draw(java.awt.Graphics g):` Felrajzolja magát a paraméterként kapott grafikus objektumra.
- `+public abstract boolean on(int x, int y):` `true`-val tér vissza, ha az (x,y) pont benne van abban a területben, amire ő ki van rajzolva, egyébként `false`-al.
- `public abstract String getToolTipText():` Azt a szöveget adja vissza, amely megjelenik egy tooltip-ben, ha az objektum fölé visszük az egeret.

### 11.3.4 Graphics

- **Felelősség**

Felelőssége a grafikus megjelenítés irányítása, az ahhoz szükséges objektumok számítartása.

- **Ósosztályok**
- 

- **Interfészek**
- 

- **Attribútumok**
- `-Game game:` a játék osztály referenciaja
- `-int movePipeIC:` A kezdőértéke 0, minden cső mozgatásával kapcsolatos interakció (gombnyomás, kattintások) 1-el növeli, majd az interakció végeztével az értéke visszaáll 0-ra.

- **-int moveIC:** A kezdőértéke 0, minden, a játékos mozgatásával kapcsolatos interakció (gombnyomás, kattintások) 1-el növeli, majd az interakció végeztével az értéke visszaáll 0-ra.
- **-int changePumpDirectionIC:** A kezdőértéke 0, minden pumpa irányának megváltoztatásával kapcsolatos interakció (gombnyomás, kattintások) 1-el növeli, majd az interakció végeztével az értéke visszaáll 0-ra.

Megjegyzés: Az előző 3 változó nevében az IC rövidítés az “Interaction Counter” szót hivatott jelképezni. Amennyiben ezek közül bármelyik értéke nem 0, minden képernyőn lévő gomb inaktív, hiszen éppen egy interakció közben vagyunk.

- **Metódusok**

- **-void pull():** Lekérdezi a modelltől az aktuális állapotot és az alapján frissíti a Graphics osztály az adatait. A gombok eseménykezelői hívják meg a feladatuk elvégzése után.
- **-void drawNodes():** Meghívja a nodes összes elemén a draw() metódust.
- **-void drawEdges():** minden edges-beli elem két szomszédja közé behúz egy megfelelő színű vonalat, a cső víztartalmától függően.
- **+void movePipe():** A “Move Pipe” gomb megnyomásakor hívódik meg. Beállítja a movePipeIC értékét 1-re.
- **+void endGame():** Az “End Game” gomb megnyomásakor hívódik meg. Meghívja a game objektum end() metódusát, majd visszaugrik a menübe, ahonnan új játékot lehet kezdeni.
- **+void makeSlippery():** Meghívja a game objektum makeSlippery() metódusát, majd a pull() metódust.
- **+void makeSticky():** Meghívja a game objektum makeSticky() metódusát, majd a pull() metódust.
- **+void breakField():** Meghívja a game objektum breakField() metódusát, majd a pull() metódust.
- **+void repairField():** Meghívja a game objektum repairField() metódusát, majd a pull() metódust.
- **+void place():** Meghívja a game objektum place() metódusát, majd a pull() metódust.
- **+void pickup():** Meghívja a game objektum pickup() metódusát, majd a pull() metódust.
- **+void changePumpDirection():** A changePumpDirection gomb megnyomásakor hívódik meg. A changePumpDirectionIC értékét beállítja 1-re.
- **+void endTurn():** Meghívja a game objektum endTurn() metódusát, majd a pull() metódust, és ellenőrzi, hogy valamelyik csapat elérte-e a kellő pontszámot, ha igen akkor egy popup-ban kiírja, hogy az egyik csapat nyert, majd meghívja az endGame() metódust.
- **+void play():** A “Play” gomb megnyomásakor hívódik meg, ez a metódus indítja el a játékot és ebben is fut. Egy ciklusban ellenőrzi a
- **+void move():** A “Move” gomb megnyomásakor hívódik meg. A moveIC értékét beállítja 1-re.
- **+Node getNodeFromClick(int x, int y):** Visszaadja azt a csomópontot, ami az (x,y) pontra van felrajzolva. Ha nincs ilyen csomópont, **null**-al tér vissza.
- **+Pipe getPipeFromClick(int x, int y):** Visszaadja annak a csőnek a referenciáját, ami az (x,y) pontra van felrajzolva. Ha nincs ilyen cső, **null**-al tér vissza.
- **+void onMouseDown(int x, int y):** minden egér klikk-re meghívódik a kurzor aktuális pozíciójának x és y koordinátáját megkapva paraméterként.

### 11.3.5 DisplayNode

- **Felelősség**

A csomópont szerű objektumok (pl. **Cistern**, **Source**, **Pump**) megjelenítéséért felelős osztály.

- **Ősosztályok**

-

- **Interfészek**

-

- **Attribútumok**

- **#Rectangle container:** A felrajzolt csomópont bennfoglaló téglalap

- **Metódusok**

- **#setPlayerIcons():** Feltölti a *playerIcons* tömb elemeit a *getGameReference()* által visszaadott **Field getPlayers()** függvényével lekért játékosok listájából. A játékosok az objektumon sorrendben a konténerben (**container**) a következő helyekre kerülnek, és minden két dimenzióban fele akkorák mint a kirajzolt csomópont:
  - 1. bal felül
  - 2. jobb felül
  - 3. bal alul
  - 4. jobb alul
- **+on(int x, int y):** true, ha az (x, y) pont benne van a container téglalapban, false, ha nincs.
- **+link(DisplayPipe edge, DisplayNode linkTowards):** Beállítja a linkelési irányt a *linkTowards* csomópont felé, amennyiben az **edge** teljesít valamilyen feltételt. A linkelési

### 11.3.6 PlayerIcon

- **Felelősség**

Egy játékost reprezentál a grafikus megjelenítés szintjén.

- **Ősosztályok**

-

- **Interfészek**

-

- **Attribútumok**

- **-Rectangle container:** A tartalmazó konténer
- **-Color color:** A karakter színe
- **-Player p:** A reprezentált játékos

- **Metódusok**

- **+void draw():** Definiálja a karakter kirajzolásához szükséges működést

### 11.3.7 DisplayPipe

- **Felelősség**

Egy kirajzolt csövet (**Pipe**) jelenít meg két csomópont között. Amennyiben a cső egyik vége nincs sehova kötve, egy létrehozáskor definiált helyre (**Rectangle**) fogja kötni azt a végét.

- **Ősosztályok**

DisplayNode

- **Interfészek**

-

- **Attribútumok**

- **-Pipe p:** A reprezentált cső.
- **-Polygon polygon:** A kirajzolt sokszög (egy változó orientációjú, **displayWidth** széles téglalap a két **DisplayNode** vagy egy **DisplayNode** és egy **Rectangle** között).
- **-float length:** A két összekötött pont közötti távolság.
- **-DisplayNode n1:** A cső egyik végét hova kell kötni.
- **-DisplayNode n2:** A cső másik végét hova kell kötni.
- **-Rectangle unboundEndConnection:** Ha **n1** és **n2** közül pontosan az egyik **null**, akkor a csőnek az oda kötendő végét ennek a téglalapnak a közepére fogja kötni. Amennyiben ilyen esetben ez is **null**, a véget a (0,0) pontba köti.
- **-Color waterColor:** A cső belsejének színe, ha van benne víz.
- **-Color noWaterColor:** A cső belsejének színe, ha nincs benne víz.
- **-Color outlineColor:** A cső körvonalának színe.
- **-float displayWidth:** A kirajzolt cső szélessége.

- **Metódusok**

- **-void calculatePolygon():** Kiszámolja a kirajzolt sokszög helyét. A cső két vége a következő helyeken lesz, a következő kritériumok alapján:
  - **n1 == null** és **n2 == null:** A **polygon** null marad, nem történik rajzolás
  - **n1 != null** és **n2 != null:** A két összekötendő pont: **n1.getContainer()** közepe és **n2.getContainer()** közepe
  - **n1** és **n2** közül pontosan az egyik **null** és **unBoundEndConnection != null:** A két összekötendő pont: az **unBoundEndConnection** téglalap közepe és a nem **null** **n\_r.getContainer()** közepe
  - **n1** és **n2** közül pontosan az egyik **null** és **unBoundEndConnection == null:** A két összekötendő pont: az (0,0) pont és a nem **null** **n\_r.getContainer()** közepe

**11.3.8-Vec2 getTriangleClosestPosition(Vec2 dir):** A paraméterként kapott **dir** irányba előállít egy pontot, am

### 11.3.9 DisplayPump

- **Felelősség**

Egy kirajzolt pumpát (**Pump**) reprezentál.

- **Ősosztályok**

DisplayNode

- **Interfészek**

-

- **Attribútumok**

- **-Pump p:** Referencia az objektumhoz tartozó pumpa objektumhoz, amit az objektum reprezentál grafikusan

- **Metódusok**

•

### 11.3.10 DisplaySource

- **Felelősség**

Egy kirajzolt forrást (**Source**) reprezentál.

- **Ősosztályok**

DisplayNode

- **Interfészek**

-

- **Attribútumok**

- **-Source s:** Referencia az objektumhoz tartozó forrás objektumhoz, amit az objektum reprezentál grafikusan
- **-Color c:** A forrás megjelenítési színe

- **Metódusok**

- **+void draw(java.awt.Graphics g):** Kirajzolja a paraméterként kapott grafikus objektumra a forrást, a **container** bennfoglaló négyzetbe, **color** kitöltési és keret színnel.
- **+Field getGameReference():** Visszaadja a reprezentált forrás objektumot (**s**).
- **+String getToolTipText():** A következő sztringet adja vissza: “**Source**”.

### 11.3.11 Map

- **Felelősség**

A pályaképet grafikusan megjelenítő és azzal interakciókat megvalósító panel.

- **Ősosztályok**

JPanel

- **Interfészek**

-

- **Attribútumok**

- **-int sizeX:** A panelen megjelenő pálya szélessége.
- **-int sizeY:** A panel megjelenő pálya magassága.
- **-int nodeWidth:** A panelen megjelenő csomópontok szélessége.
- **-int nodeWidth:** A panelen megjelenő csomópontok magassága.
- **-DisplayNode draggingNode:** Amennyiben a felhasználó éppen Drag&Drop operációt végez, a mozgatott csomópont referenciája, egyébként **null**.
- **#List<DisplayNode> nodes:** A kirajzolt csomópontok listája.
- **#List<DisplayPipe> edges:** A kirajzolt élek (csövek) listája.

- **Metódusok**

- **+void drawEdges(List<Pipe> pipes):** A kapott *pipes* lista alapján létrehozza a hozzájuk tartozó *DisplayNode* elemeket és eltárolja őket az *edges* listában. A létrehozás minden *pipes-belli p* csőre a következők szerint:  
*p.getNeighbors() == 0:* A csövet nem veszi figyelembe  
*p.getNeighbors() == 1:* A cső egyik szomszédja azon *nodes*-beli *n*, melyre *n.getGameReference() == p.GetNeighbors.get(0)*. A csőnek át kell adni egy olyan négyzetet, ahova a másik véget kósse, ezt *n1* közepétől kicsit lejjebb számolja ki.  
*p.getNeighbors() == 1:* A cső egyik szomszédja azon *nodes*-beli *n1*, melyre *n1.getGameReference() == p.GetNeighbors.get(0)*, másik szomszédja azon *nodes*-beli *n2*, melyre *n2.getGameReference() == p.GetNeighbors.get(1)*.
- **+DisplayNode getNodeFromClick(int x, int y):** Visszaadja azt a *nodes*-beli elemet, ami az (x,y) pontra van felrajzolva. Ha több is ide van felrajzolva, ezek közül az egyiket, ha egy sincs, *null*-t.
- **+DisplayPipe getPipeFromClick(int x, int y):** Visszaadja azt az *edges*-beli elemet, ami az (x,y) pontra van felrajzolva. Ha több is ide van felrajzolva, ezek közül az egyiket, ha egy sincs, *null*-t.
- **+void pumpPlaced(List<Pipe> pipes, List<Pump> pumps):** A kapott két listából kitalálja, hogy melyik az új lerakott pumpa, melyik az eltűnt cső és melyik a két új cső. Ezután újrarájzolja a csöveket (*drawEdges(pipes)*).
- **+void mousePressed(MouseEvent e):** Egér lenyomásakor hívódik meg, ha a jobb kíkk lett lenyomva, és van a kíkk (x,y) koordinátáján egy *DisplayNode*, beállítja azt a *draggingNode*-ra. (Ennek kiderítéséhez: *getNodeFromClick(e.getX(), e.getY())*).
- **+void mouseReleased(MouseEvent e):** Egérgomb felengedésekor hívódik meg, ha a jobb kíkk lett felengedve, a *draggingNode null* lesz.
- **+void mouseDragged(MouseEvent e):** Egérgomb lenyomott állapota mellett egérmozgásra hívódik meg, amennyiben a *draggingNode* nem *null*, annak konténerének (*getContainer()*) x és y koordinátáját a kurzor x és y koordinátájára állítja.
- **+String getToolTipText(MouseEvent event):** Felülírja a panel azonos szignatúrájú függvényét, az event-ből az egér koordinátái alapján kideríti (*getPipeFromClick(..)* ill. *getNodeFromClick(..)*), melyik felrajzolt objektum felett van a kurzor, és annak megjeleníti a tooltip-jét (*getToolTipText()*).
- **+void paint(java.awt.Graphics g):** Felrajzolja először az éleket (*edges*) aztán a csúcsokat (*nodes*), azok *draw(g)* metódusának meghívásával

### 11.3.12 Vec2

- **Felelősség**

Kétdimenziós vektorok reprezentációja és azok műveleteinek megvalósítása.

- **Ósosztályok**

-

- **Interfészek**

-

- **Attribútumok**

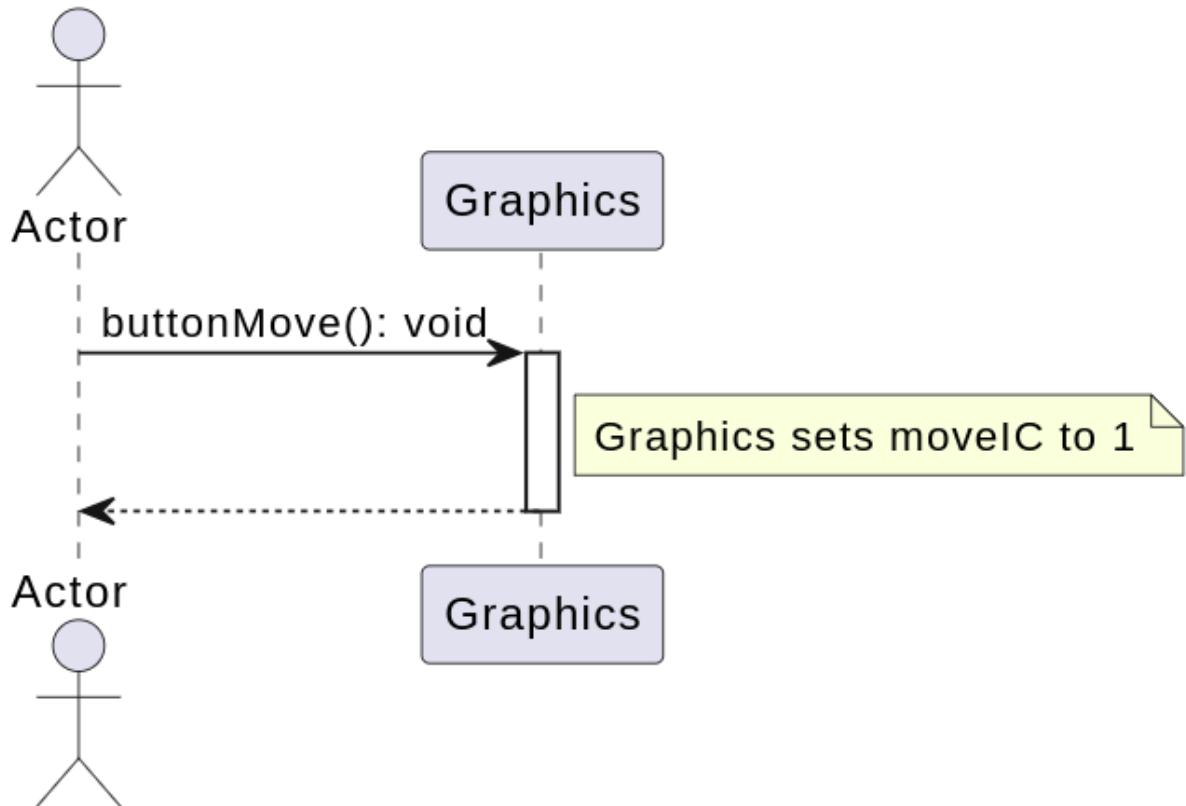
- **+float x:** A vektor x koordinátája.
- **+float y:** A vektor y koordinátája.

- **Metódusok**

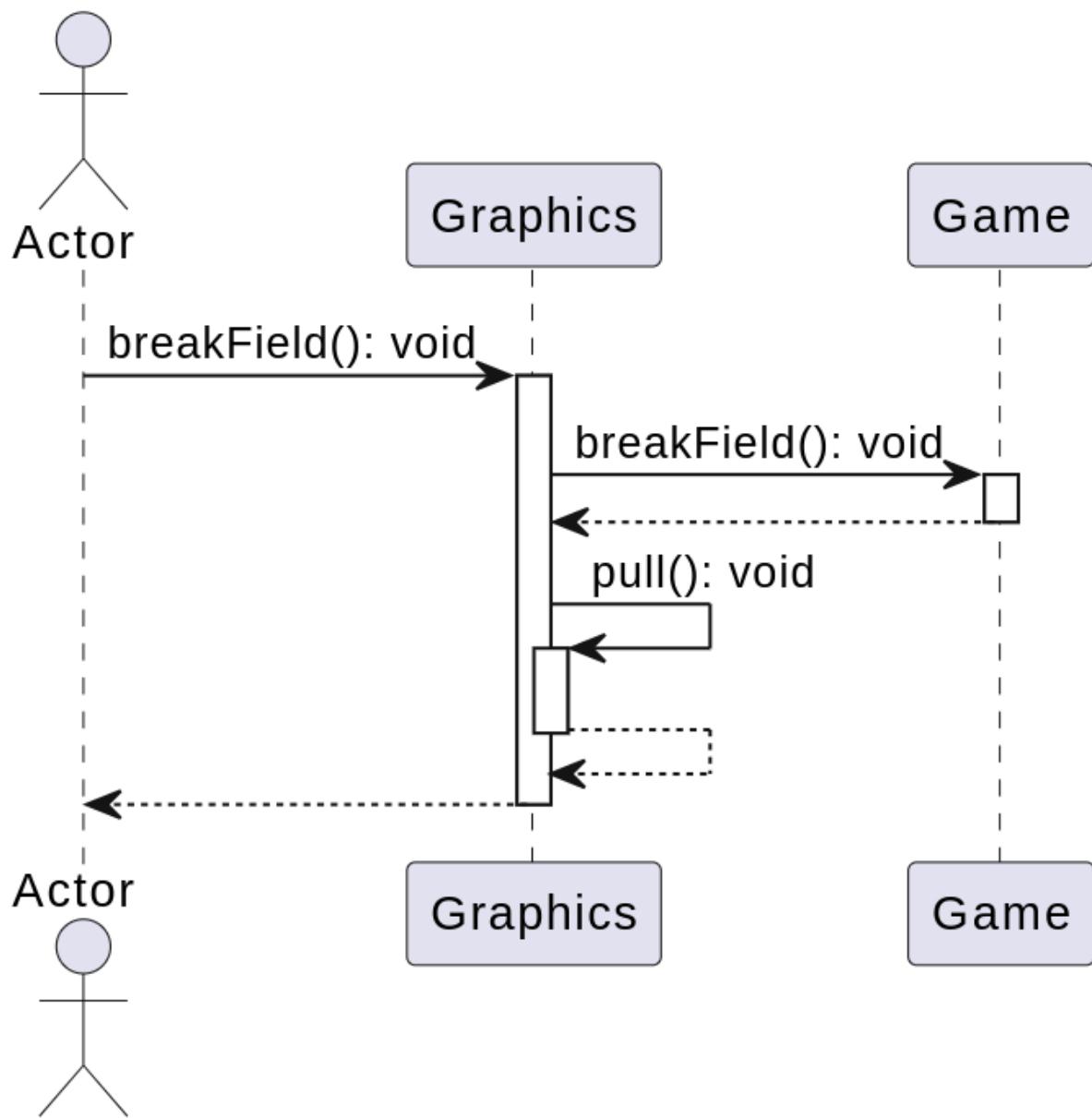
- **+int ix():** Visszaadja az x koordinátát egész számra kerekítve.
- **+int iy():** Visszaadja az y koordinátát egész számra kerekítve.
- **+static float dot(Vec2 v1, Vec2 v2):** Skaláris szorzás.
- **+static Vec2 add(Vec2 v1, Vec2 v2):** Vektorok összeadása.
- **+static Vec2 subtract(Vec2 v1, Vec2 v2):** Vektorok kivonása.
- **+static float length(Vec2 v):** Vektor hosszának meghatározása.
- **+static Vec2 multiply(Vec2 v, float s):** Vektor szorzása skalárral.
- **+static Vec2 divide(Vec2 v, float s):** Vektor osztása skalárral.
- **+static Vec2 normalize(Vec2 v):** Vektor normalizálása.

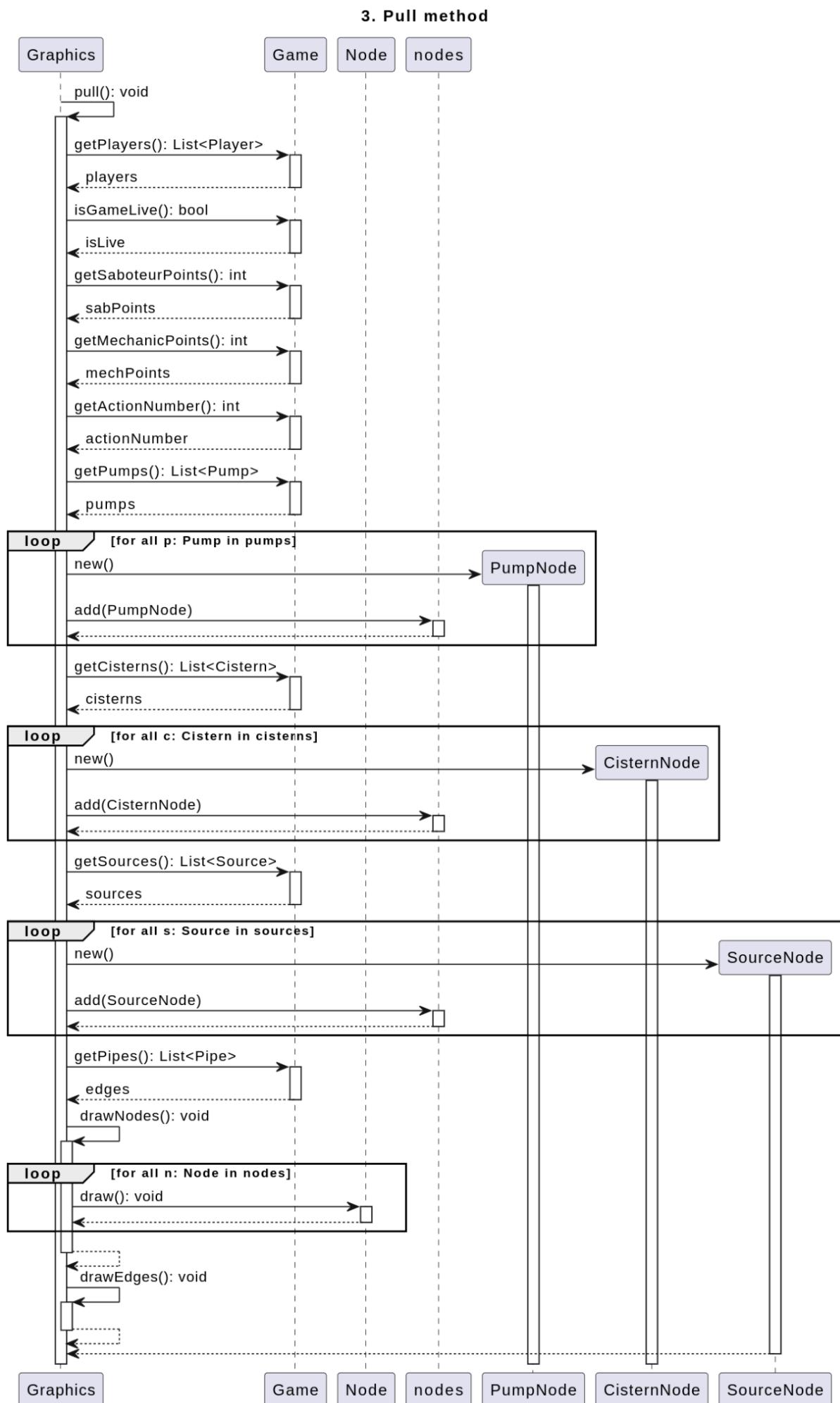
#### **11.4 Kapcsolat az alkalmazói rendszerrel**

## 1. Button Move

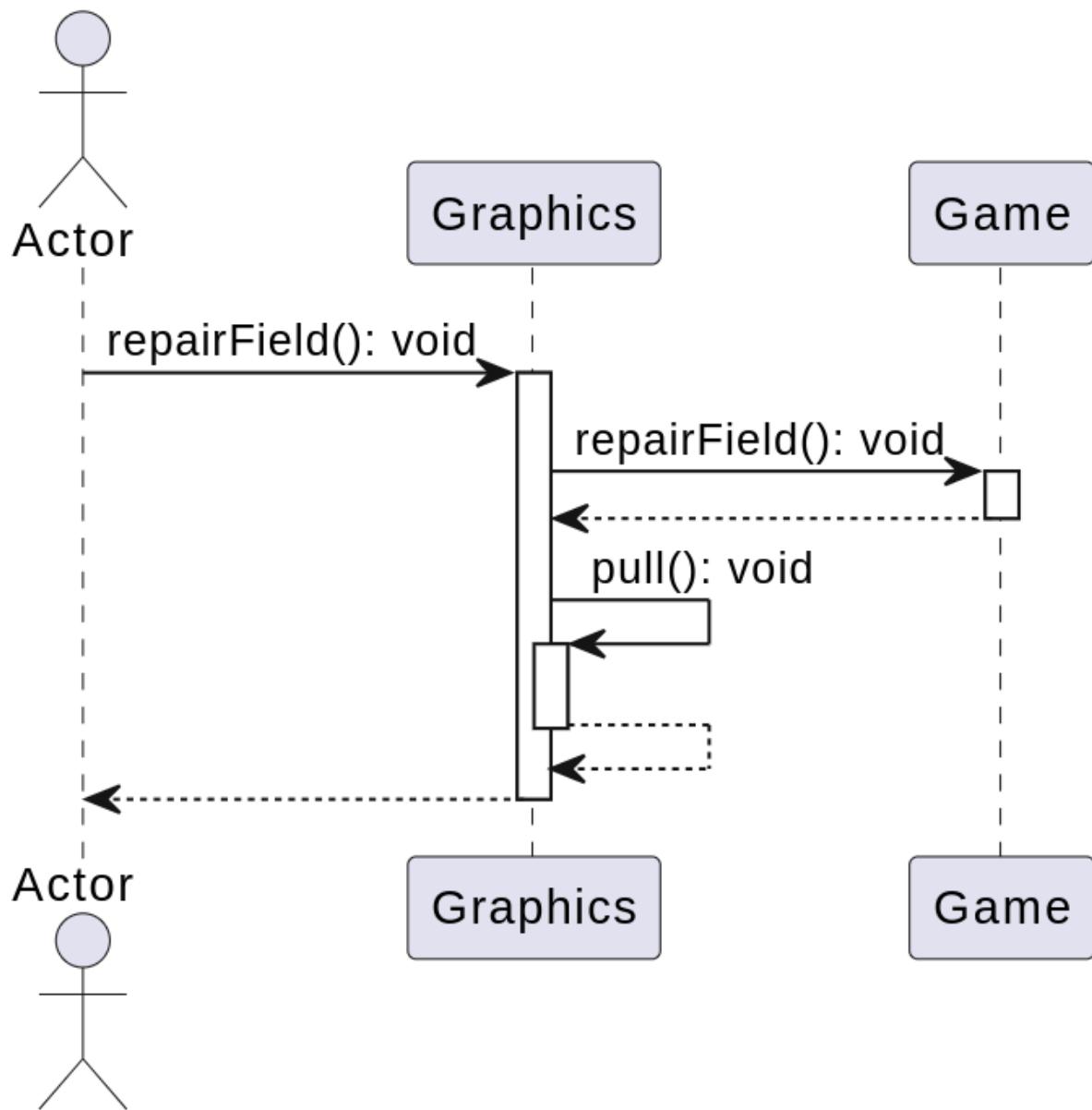


## 2. Button Break

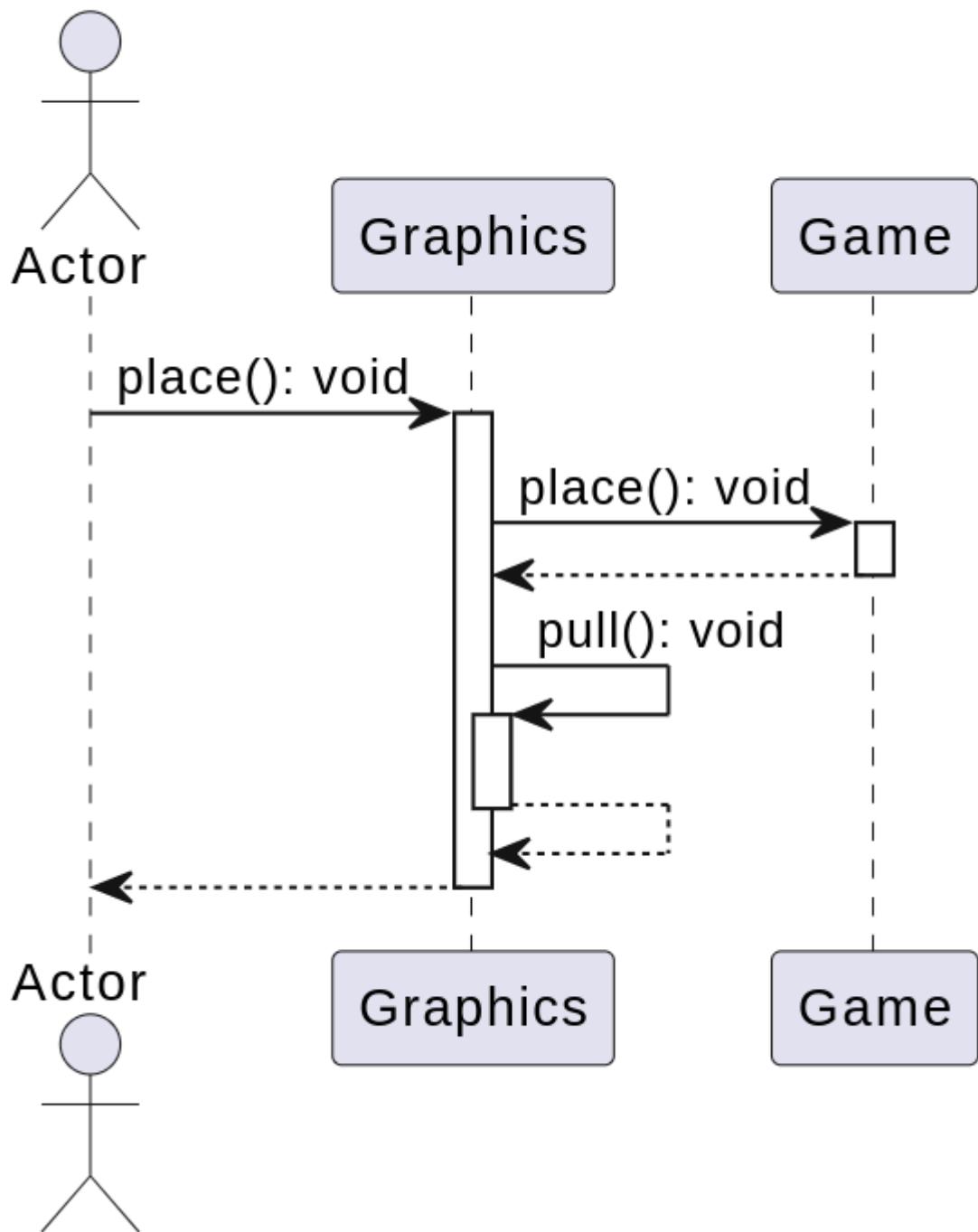




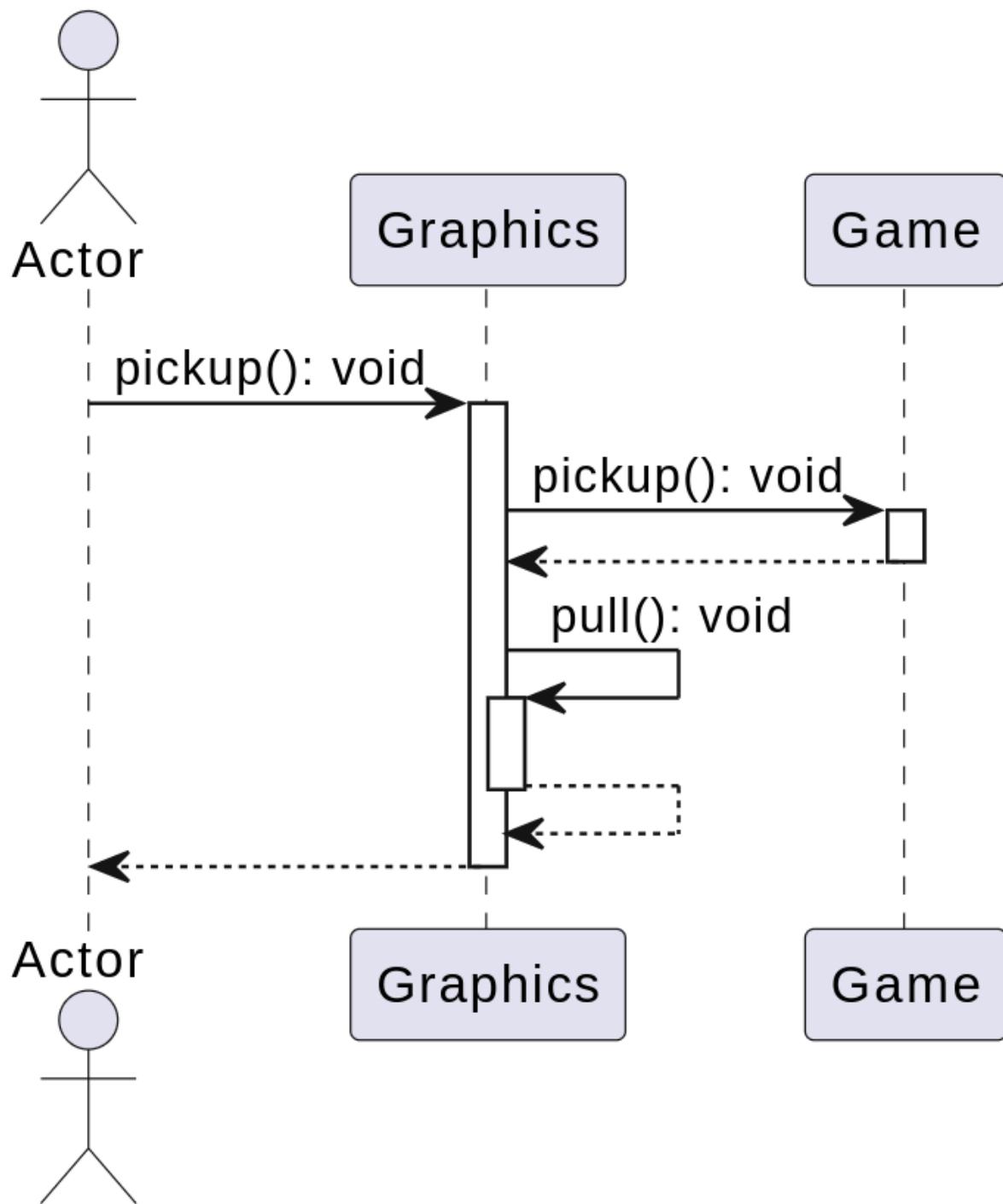
## 4. Button Repair



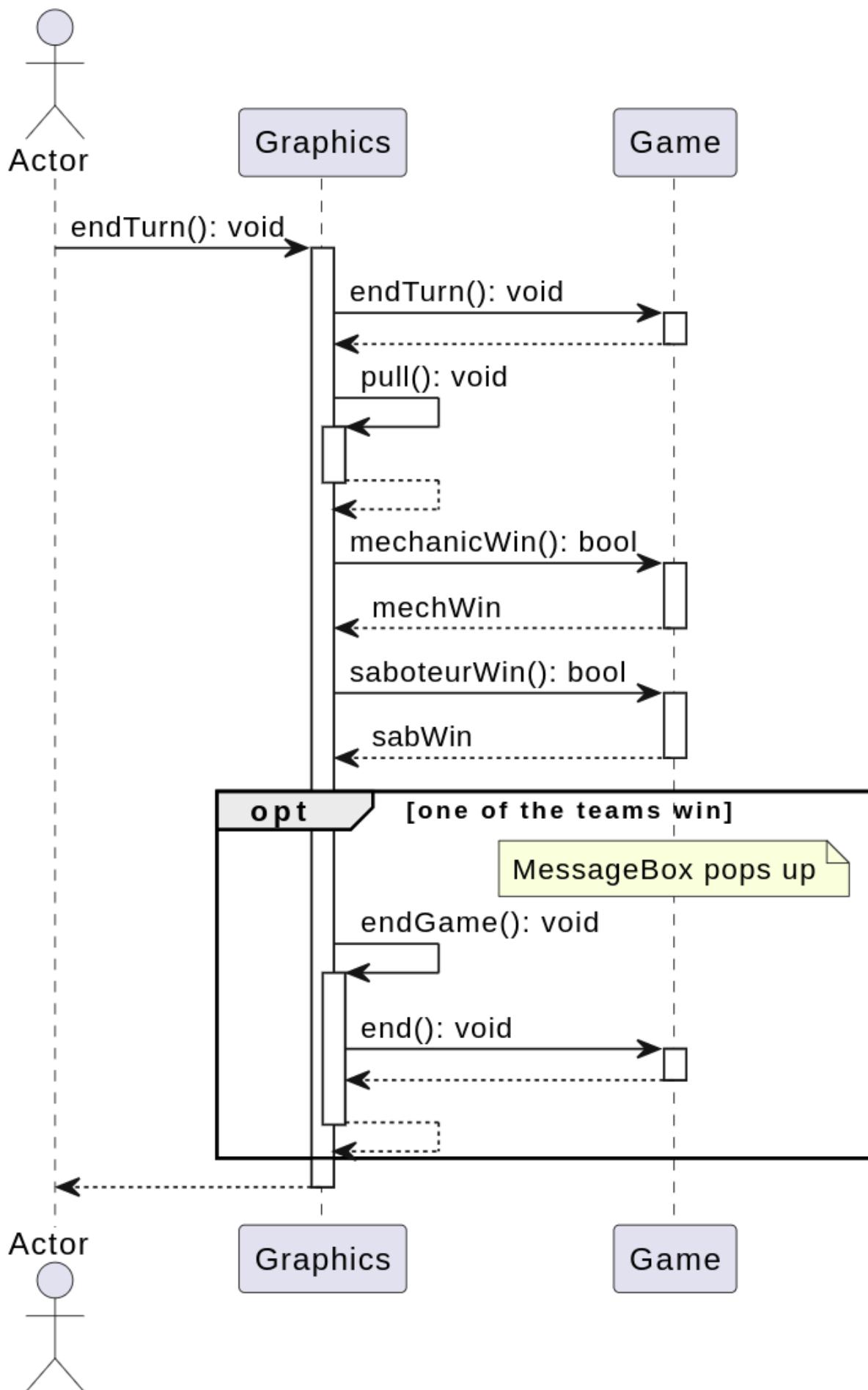
## 5. Button Place



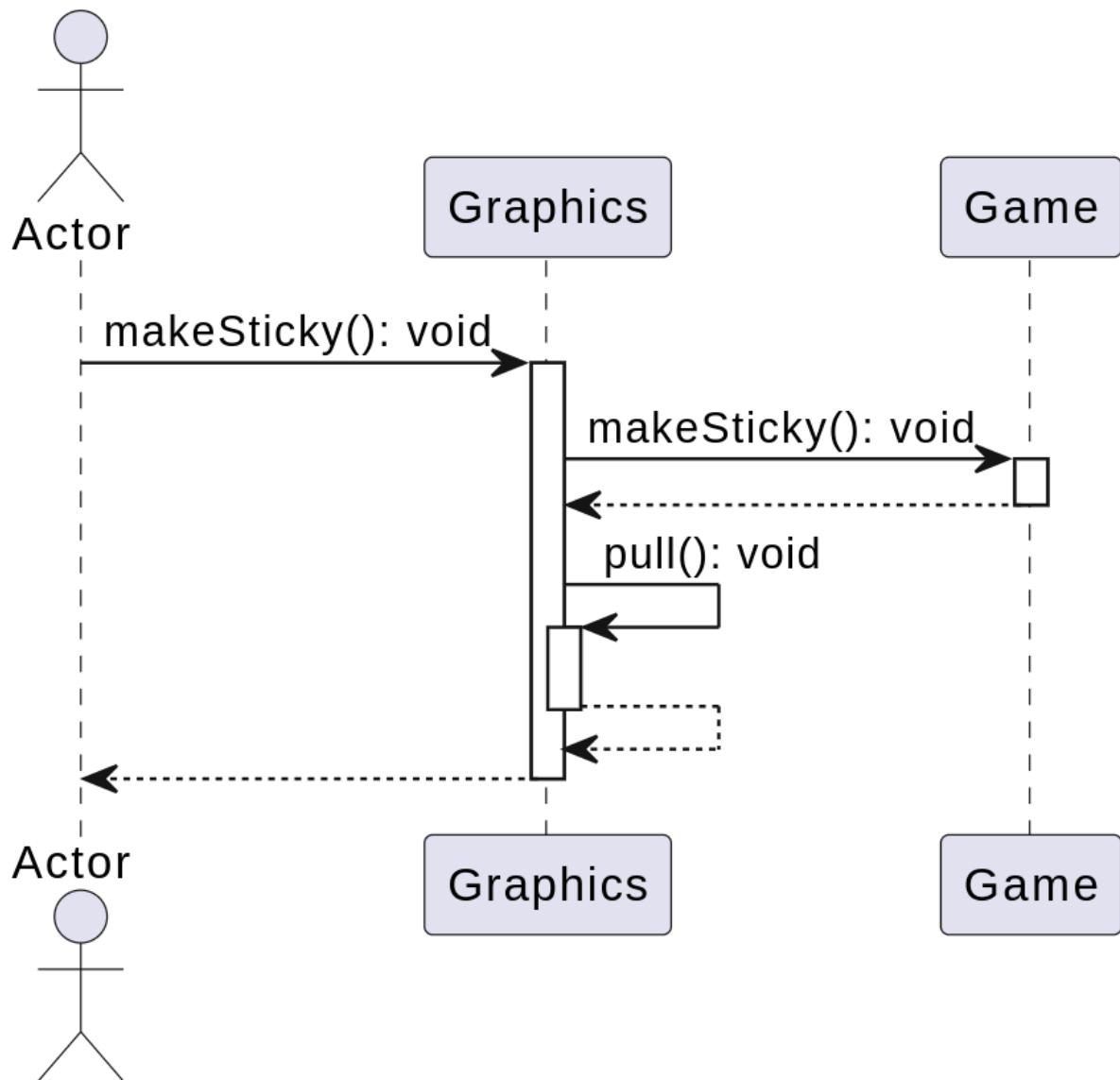
## 6. Button Pickup



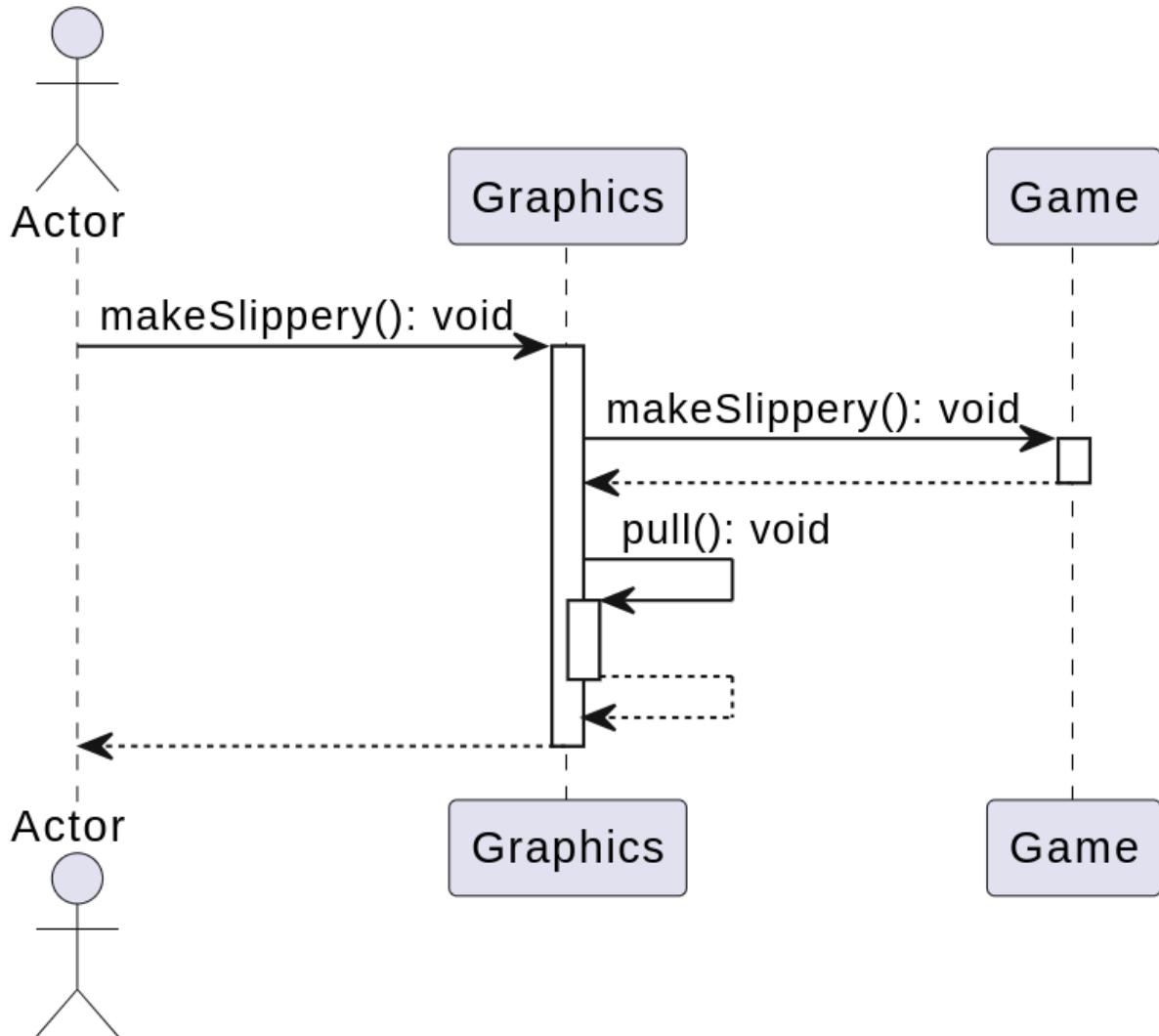
## 7. Button EndTurn



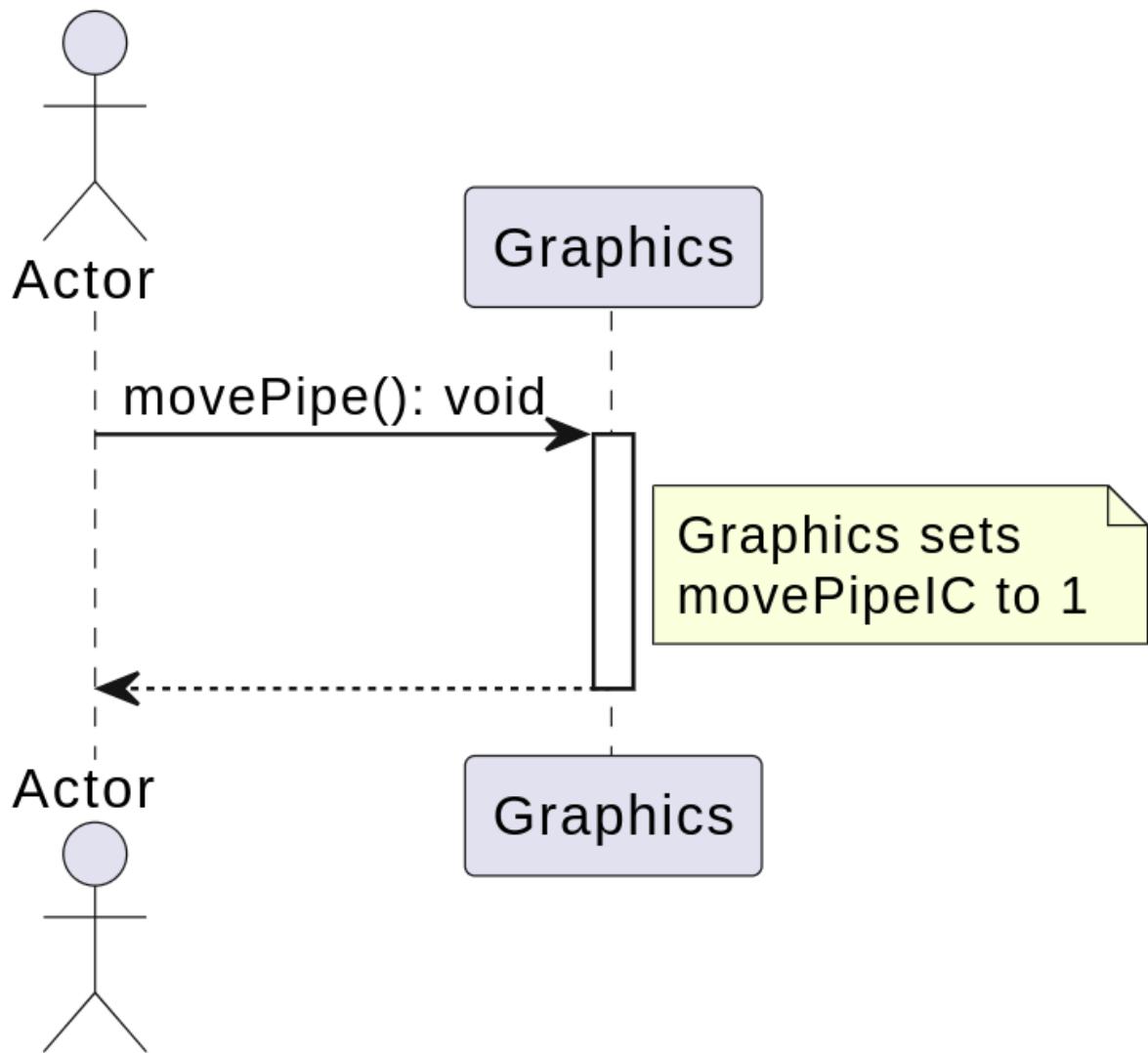
## 8. Button MakeSticky



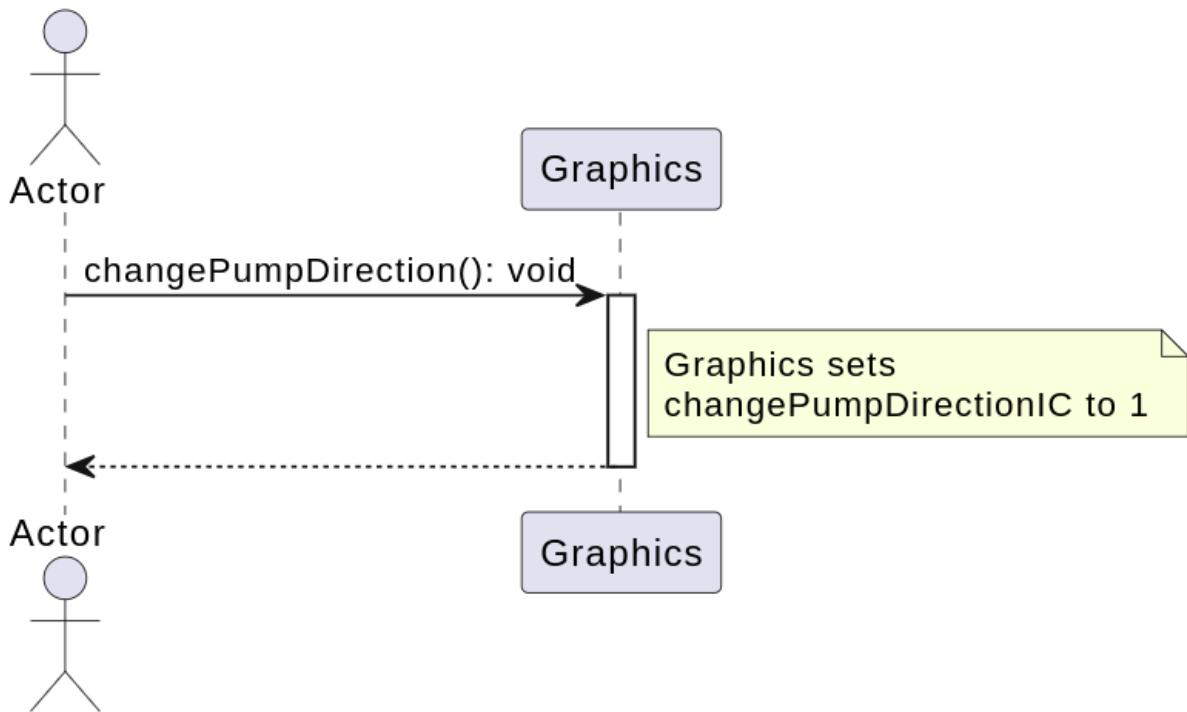
## 9. Button MakeSlippery



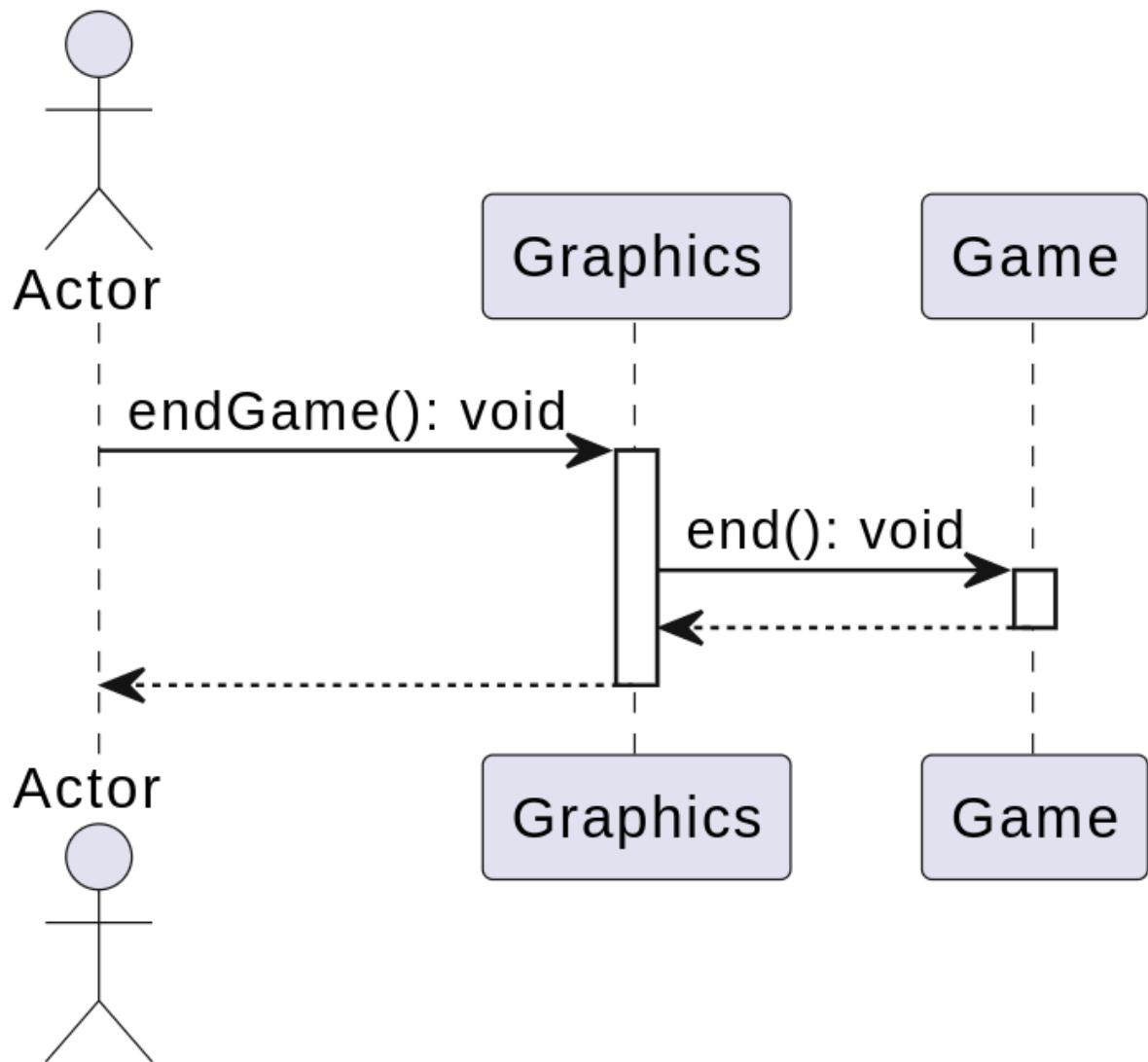
## 10. Button MovePipe



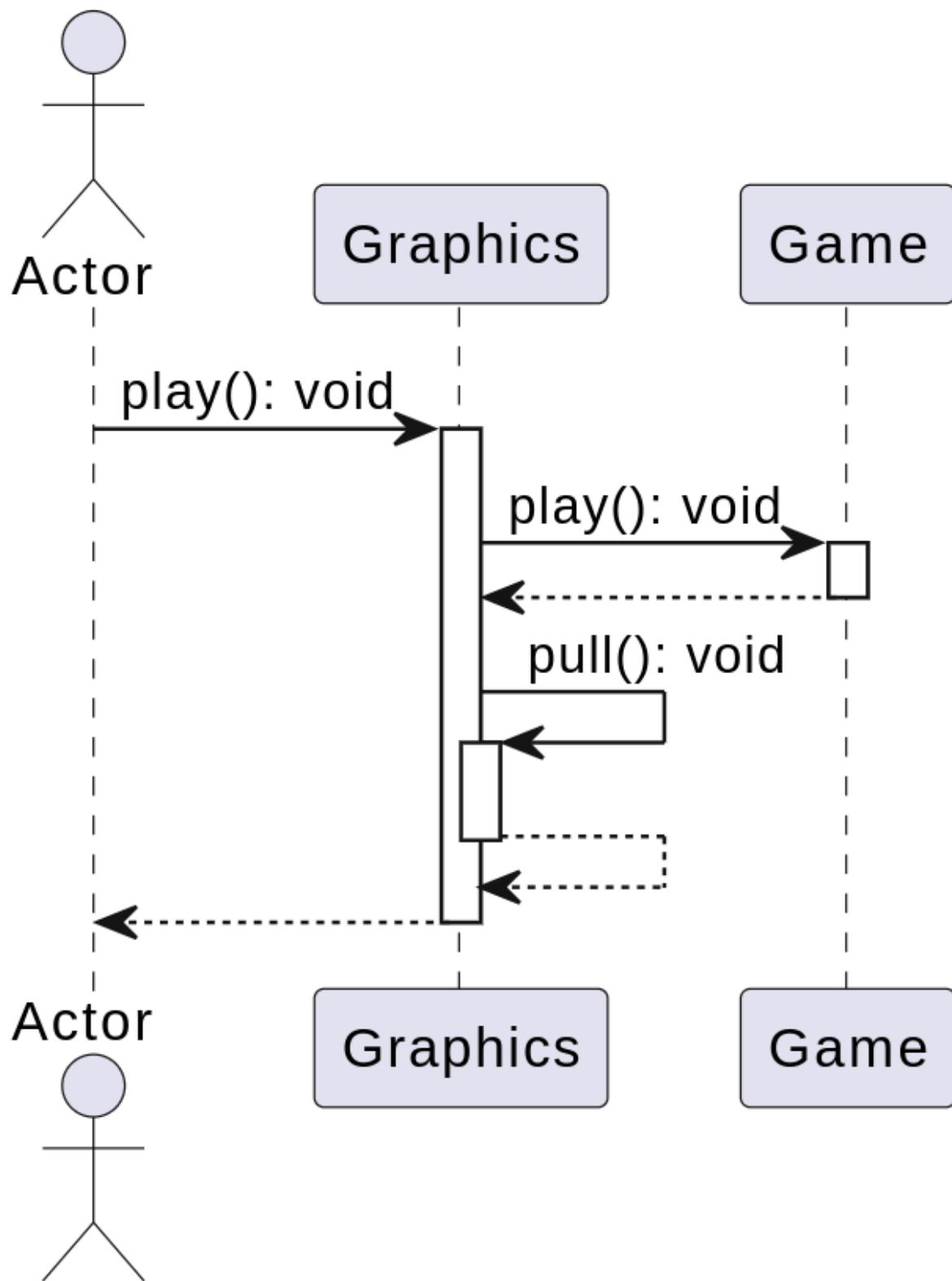
## 11. Button ChangePumpDirection



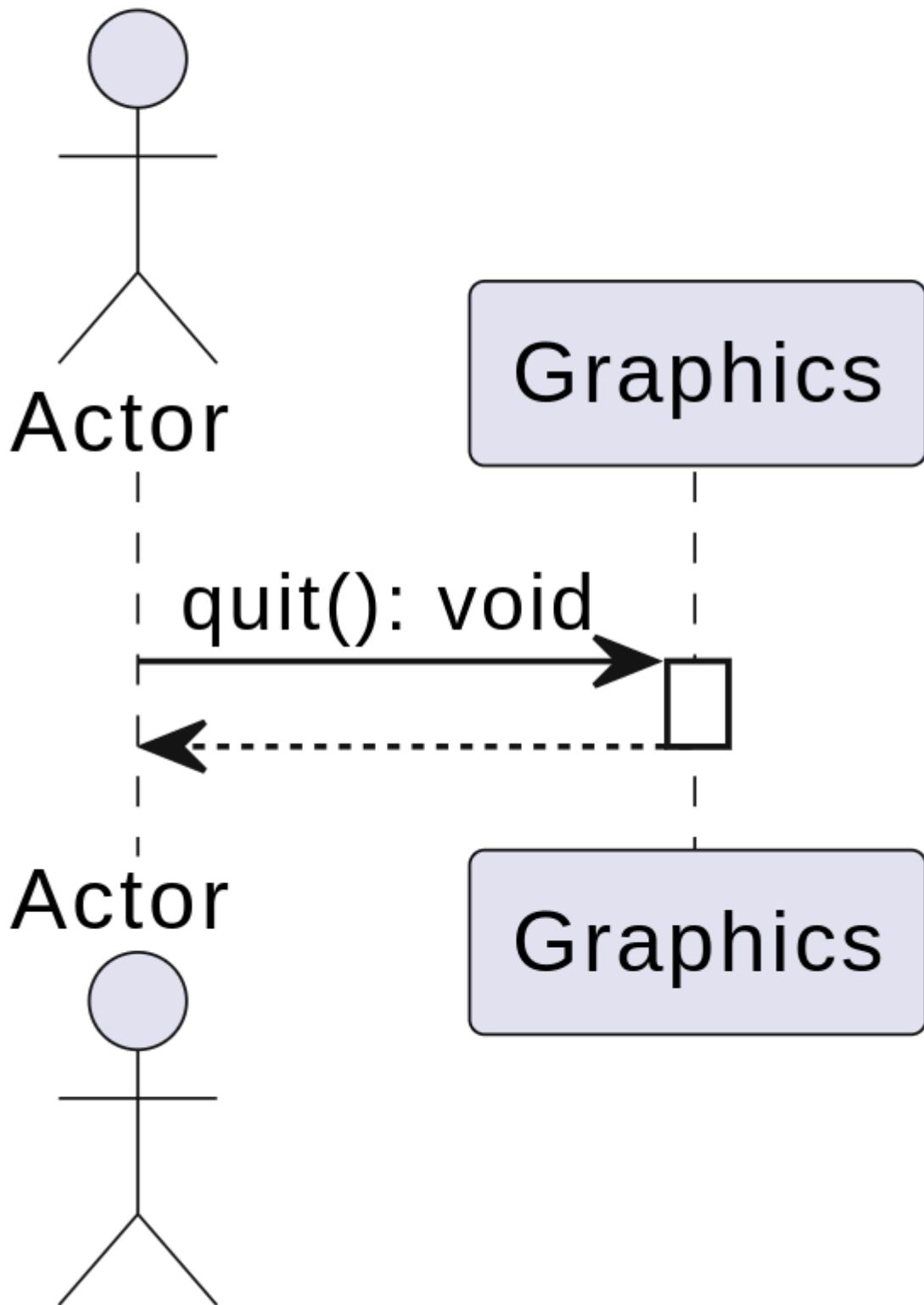
## 12. Button EndGame



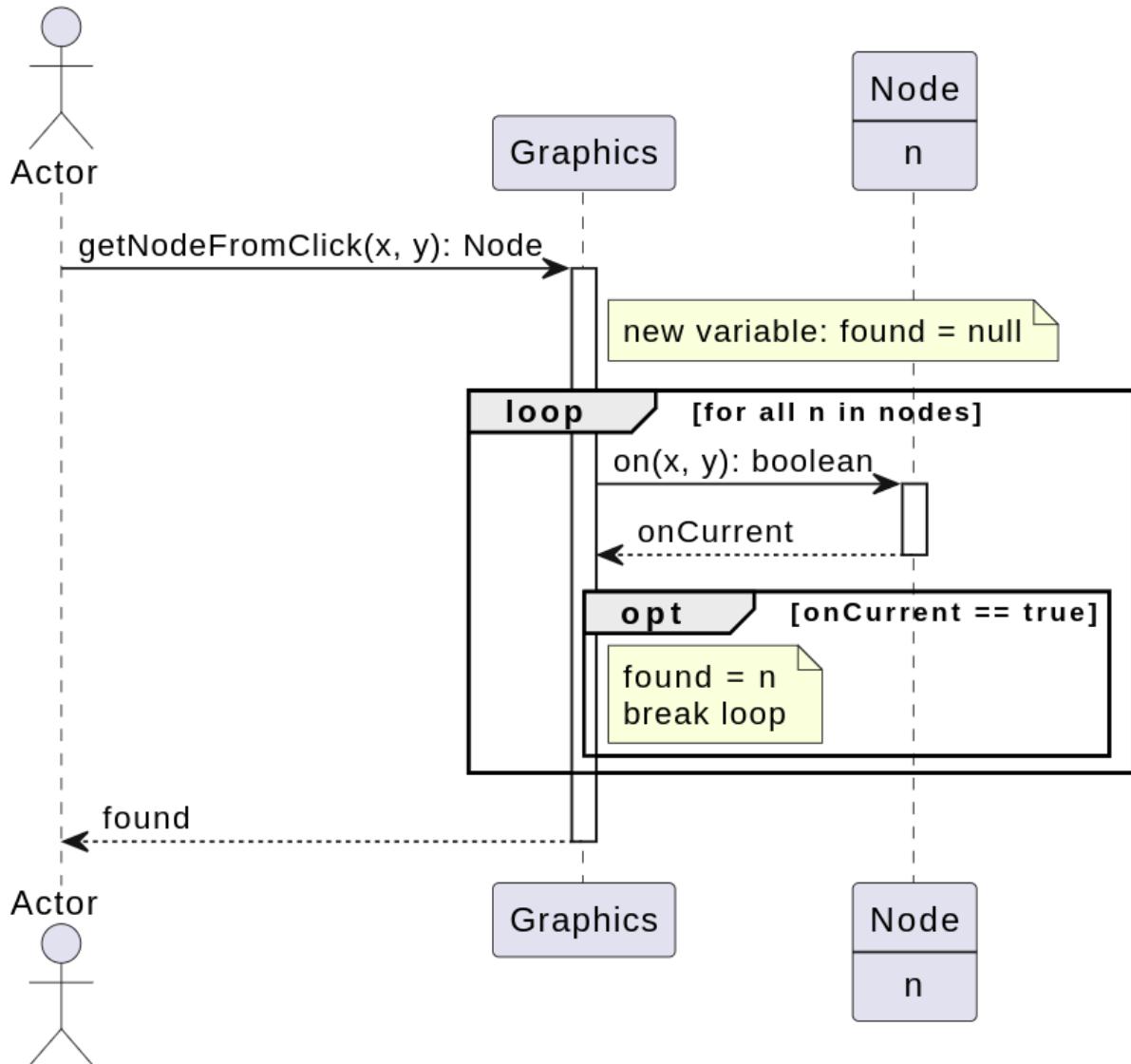
## 13. Button Play



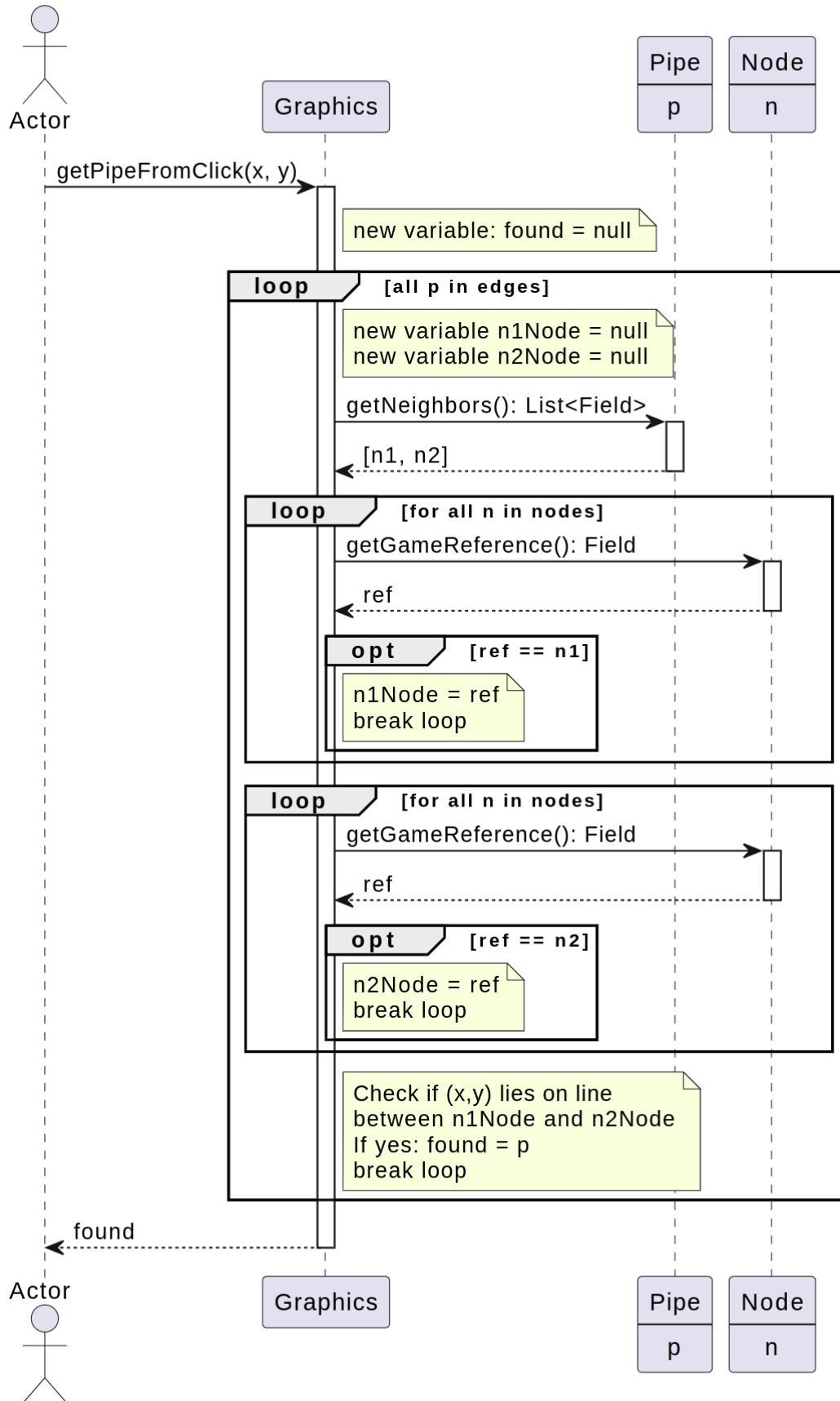
# 14. Button Quit

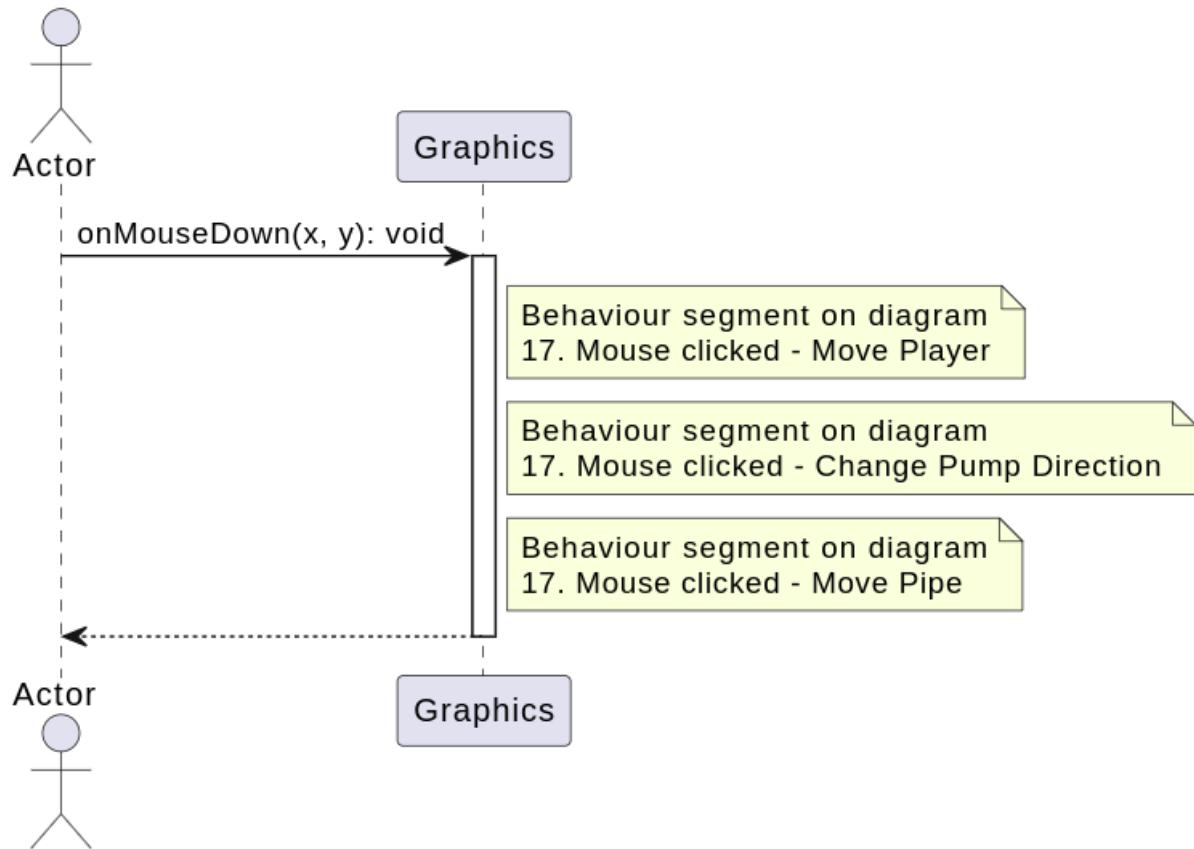


## 15. Get Node From Click

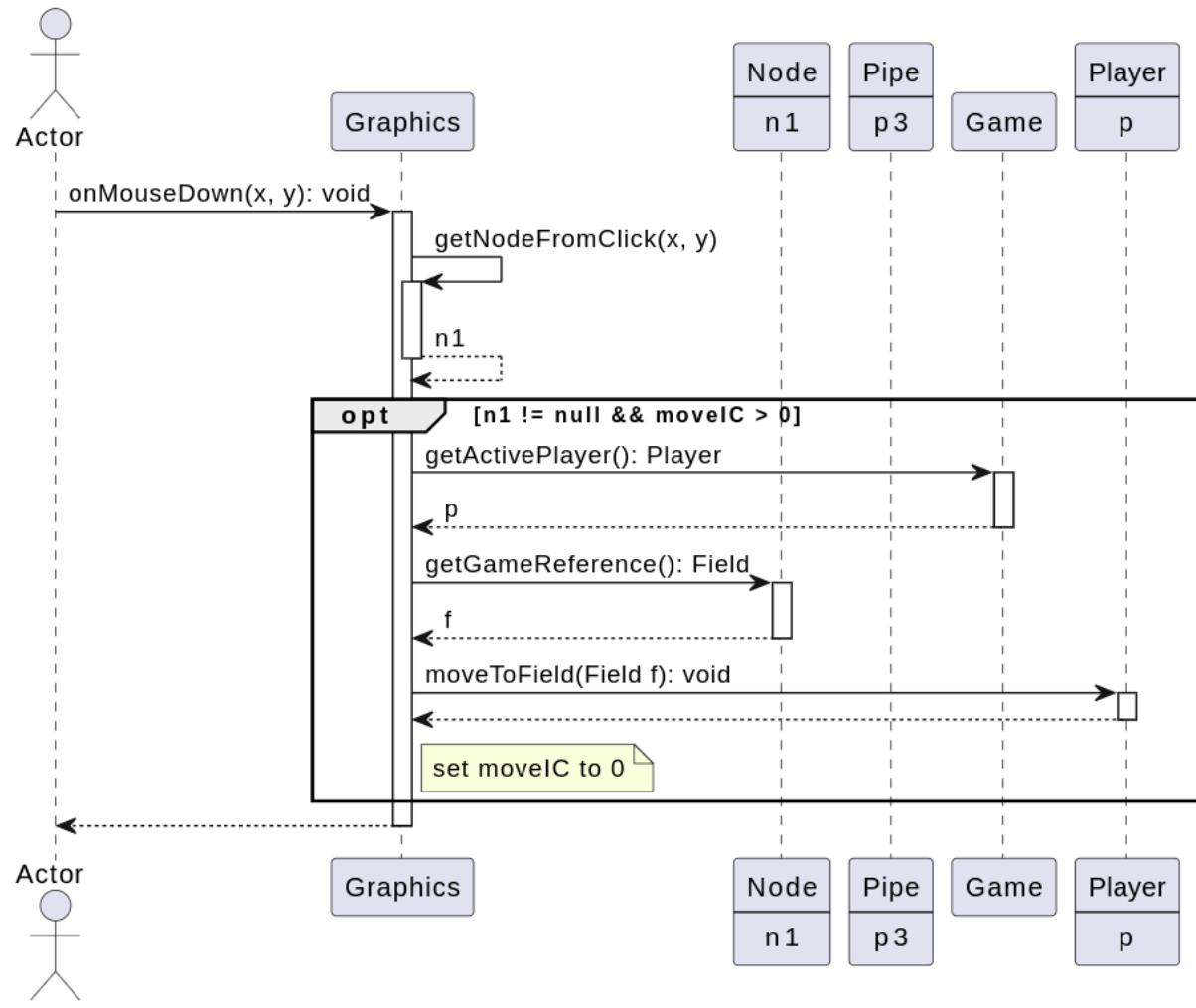


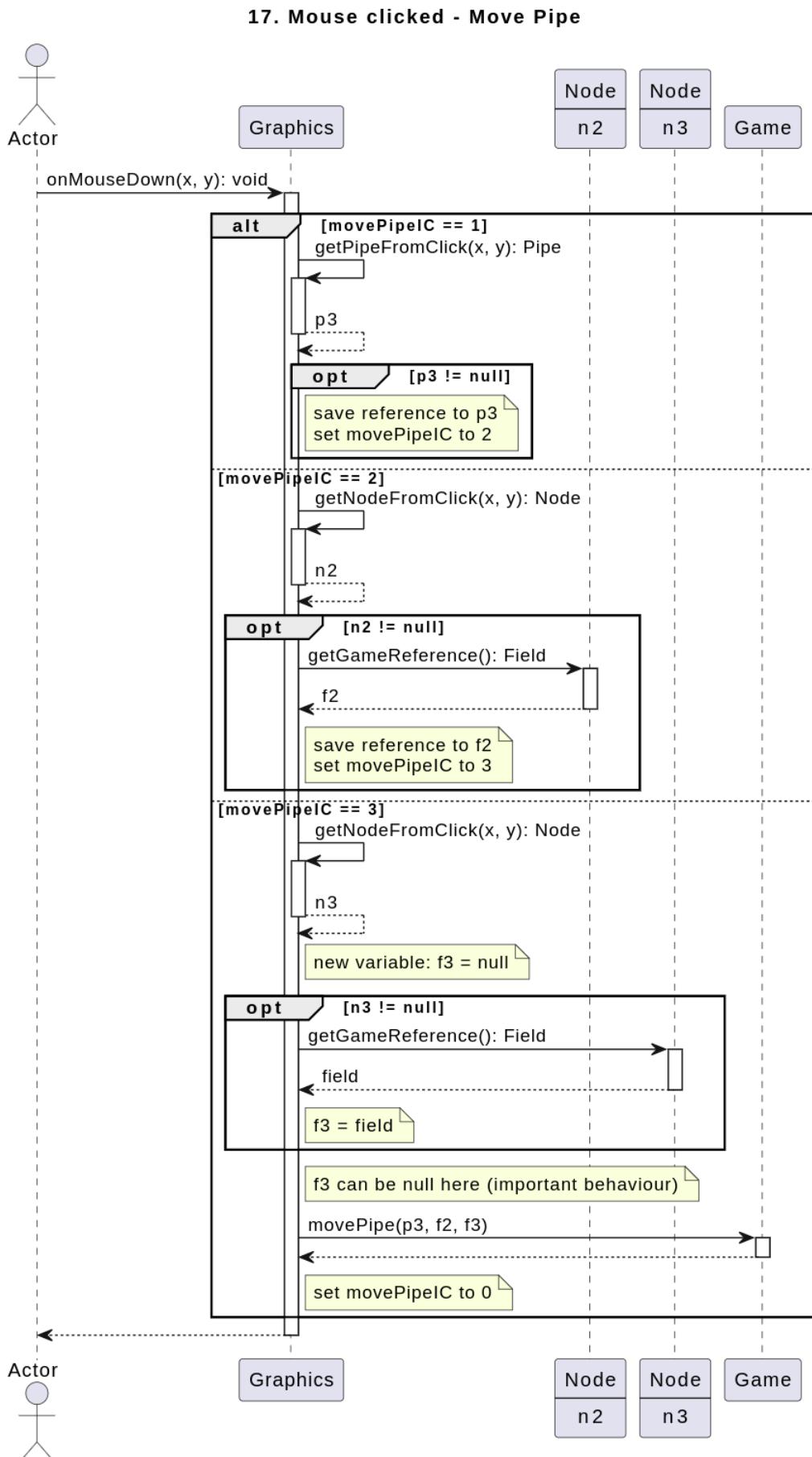
### 16. Get Pipe From Click



**17. Mouse clicked - Abstract**

## 17. Mouse clicked - Move Player





## 11.5 Napló

Kezdet	Időtartam	Résznevők	Leírás
2023.05.20 10:00	2.5 óra	Garai Mali Varga Zavadil	Értekezlet. Grafikus felület terveinek egyeztetése, feladatok kiosztása.
2023.05.20 20:00	4 óra	Zavadil	11.1 megírása, osztályok terveinek kiegészítése
2023.05.21. 8:30	1 óra	Varga	11.2 megírása
2023.05.21. 14:00	2 óra	Völgyesi	Osztálydiagram elkészítése
2023.05.21. 17:00	3 óra	Mali	11.3 megírása
2023.05.21. 21:00	4 óra	Zavadil	Grafikus felületen az egyes objektumok kiválasztásának tervezése, osztálydiagram hibák javítása, szekvenciadiagramok írása, dokumentum formázása, képek beillesztése
2023.05.21. 21:00	1 óra	Mali	
2023.05.21. 14:30	2,5 óra	Garai	Szekvenciadiagramok írása

## 13. Grafikus változat beadása

### 13.1 Fordítási és futtatási útmutató

#### 13.1.1 Fájllista

Fájl neve	Méret (byte)	Keletkezés ideje	Tartalom
Cistern.java	3066	May 31 00:11	A megegyező nevű osztály definíciója
DisplayCistern.java	1439	May 29 13:57	A megegyező nevű osztály definíciója
DisplayField.java	1603	May 31 00:11	A megegyező nevű osztály definíciója
DisplayNode.java	4198	May 31 00:11	A megegyező nevű osztály definíciója
DisplayPipe.java	8153	May 31 00:11	A megegyező nevű osztály definíciója
DisplayPump.java	8565	May 29 13:57	A megegyező nevű osztály definíciója
DisplaySource.java	1290	May 29 13:57	A megegyező nevű osztály definíciója
Field.java	3632	May 29 13:57	A megegyező nevű osztály definíciója
Game.java	9251	May 31 00:11	A megegyező nevű osztály definíciója
Graphics.java	23262	May 31 00:11	A megegyező nevű osztály definíciója
GraphicsDesigner.form	15347	May 29 13:57	A grafikus elrendezések definíciója
GraphicsDesigner.java	13870	May 29 13:57	A megegyező nevű osztály definíciója
Main.java	41450	May 31 00:11	A megegyező nevű osztály definíciója
Map.java	11926	May 31 00:11	A megegyező nevű osztály definíciója
Mechanic.java	3860	May 31 00:11	A megegyező nevű osztály definíciója
Periodic.java	380	May 29 13:57	A megegyező nevű osztály definíciója
Pipe.java	8205	May 31 00:11	A megegyező nevű osztály definíciója

Player.java	2655	May 31 00:11	A megegyező nevű osztály definíciója
PlayerIcon.java	1401	May 29 13:57	A megegyező nevű osztály definíciója
Pump.java	4829	May 31 00:11	A megegyező nevű osztály definíciója
Saboteur.java	903	May 31 00:11	A megegyező nevű osztály definíciója
Source.java	1100	May 31 00:11	A megegyező nevű osztály definíciója
Stateful.java	254	May 29 13:57	A megegyező nevű osztály definíciója
Timer.java	2734	May 29 13:57	A megegyező nevű osztály definíciója
Vec2.java	2415	May 29 13:57	A megegyező nevű osztály definíciója

### 13.1.2 Fordítás és telepítés

Egy az összes 13.1.1-ben felsorolt fájl-t tartalmazó könyvtárba lépjünk be. Ezután ellenőrizzük, hogy a java és javac parancsok fel vannak-e telepítve, majd adjuk ki a következő parancsot a fordításhoz:

```
javac *.java
```

### 13.1.3 Futtatás

Az előző fejezetben leírt lépések után ugyanabban a könyvtárban adjuk ki a következő parancsot a futtatáshoz:

```
java Main
```

## 13.2 Értékelés

Tag neve	Tag neptun	Munka százalékban
Zavadil Balázs Dávid	FOIFDF	20
Varga Dávid	Z8JXEQ	20
Völgyesi Soma	GBI4MD	20

Garadi Donát	CRSL00	20
Mali Bence	ZA2ENI	20

### 13.3 Napló

Kezdet	Időtartam	Résznevők	Leírás
2023.05.25. 17:00	2 óra	Zavadil Völgyesi Mali Varga Garai	Értekezlet: grafikus felület logikájának átbeszélése, feladatok kiosztása
2023.05.26. 17:30	3 óra	Garai	Node osztály keretének megírása, Graphics osztály, GUI elkészítése
2023.05.27. 10:00	4 óra	Zavadil	Grafikus megjelenítés alapjai
2023.05.27. 19:00	7.5 óra	Zavadil	Vec2 osztály, pumpálási irány háromszögei, egér klikkból Node / Pipe, komponensek és állapotaiak kirajzolása, tooltip-ek, játékosok kirajzolása, komponensek áthelyezése Drag&Drop-al, pálya dinamikus frissítése
2023.05.28. 10:00	2 óra	Zavadil	Grafikus felület hibáinak javítása, modell módosítások dokumentálása, javadoc kommentek
2023.05.28. 12:30	3.5 óra	Zavadil	Módosítások dokumentálásának befejezése, javadoc kommentek, osztálydiagramok, minimális játék debug
2023.05.29. 14:00	3 óra	Völgyesi Mali	Grafikus felület tesztelése, hibák javítása, alap pálya létrehozása
2023.05.29 17:00	6 óra	Varga	A move funkció HELYES implementálása
2023.05.30 10:00	6 óra	Varga	Játék logikájának korrigálása. A folyam működésének javitása.
2023.05.30 18:00	6 óra	Varga	Maradék functiók megvalósítása
2023.05.31. 09:00	1.5 óra	Mali	Dokumentáció megírása

## 14. Összefoglalás

### 14.1 A projektre fordított összes munkaidő

Tag neve	Munkaidő (óra)
Garadi Donát	71.5
Mali Bence	58.5
Varga Dávid	73
Völgyesi Soma	50
Zavadil Balázs	114.25
<b>Összesen</b>	<b>367.25</b>

- A feltöltött programok forrássorainak száma**

Fázis	Kódsorok száma
Szkeleton	1888
Prototípus	3253
Grafikus változat	4843
<b>Összesen</b>	<b>9984</b>

### 14.2 • Projekt összegzés

#### 14.2.1 Mit tanultak a projektből konkrétan és általában?

Tapasztalatot szereztünk a csapatban dolgozás szervezéséről, a feladatok hatékony kiosztásáról. Javult az objektumorientált, és az általanosságban vett szoftvertervezési érzékünk.

#### 14.2.2 Mi volt a legnehezebb és a legkönnyebb?

A legnehezebb a különböző időbeosztásaink összeszervezése és a feladatok szétosztása volt. A legkönnyebbnek a dokumentáció megírását találtuk.

#### 14.2.3 Összhangban állt-e az idő és a pontszám az elvégzendő feladatokkal?

A többi tárgy kreditértékéhez hasonlítva sokkal több volt az egy kreditre jutó munka. Lehetne több kredites a tárgy.

#### 14.2.4 Ha nem, akkor hol okozott ez nehézséget?

Az volt a nehézség, hogy minden héten rá kellett szánni mindenkinek egész sok időt.

#### 14.2.5 Milyen változtatási javaslatuk van?

-

#### 14.2.6 Milyen feladatot ajánlanának a projektre?

-

**14.2.7 Egyéb kritika és javaslat**

-