Pozícia lopty, videnie lopty hráčom

Dôležité triedy pri určovaní pozície lopty:

DynamicObject.java - trieda slúžiaca na opis polohy a rýchlosti objektov pohybujúcich sa v priestore – lopta

WorldModel.java -opisuje stav sveta, teda ostatných hráčov, lopty,(objektov, ktorých agent vidí), odhaduje ich pozície na základe odhadnutej rýchlosti

ParsedData - Trieda obsahuje dáta, ktoré boli získané z prijatej správy od servera. Sú to dáta z receptorov (zrak, sluch, sila pôsobiaca na nohy), a tiež dáta opisujúce aktuálnu situáciu na ihrisku.

Metóda pre vypočítanie vzdialenosti medzi loptou a hráčom (pozícia lopty, ktorú berie v úvahu je nastavená vo WorldModel.

```
public double getDistanceFromBall() {
    if(this.worldModel.getBall() == null) {
        return 0;
    }
    else
        return DistanceHelper.computeDistanceBetweenObjects(worldModel.getBall().getPosition(), this.getPosition());
}
```

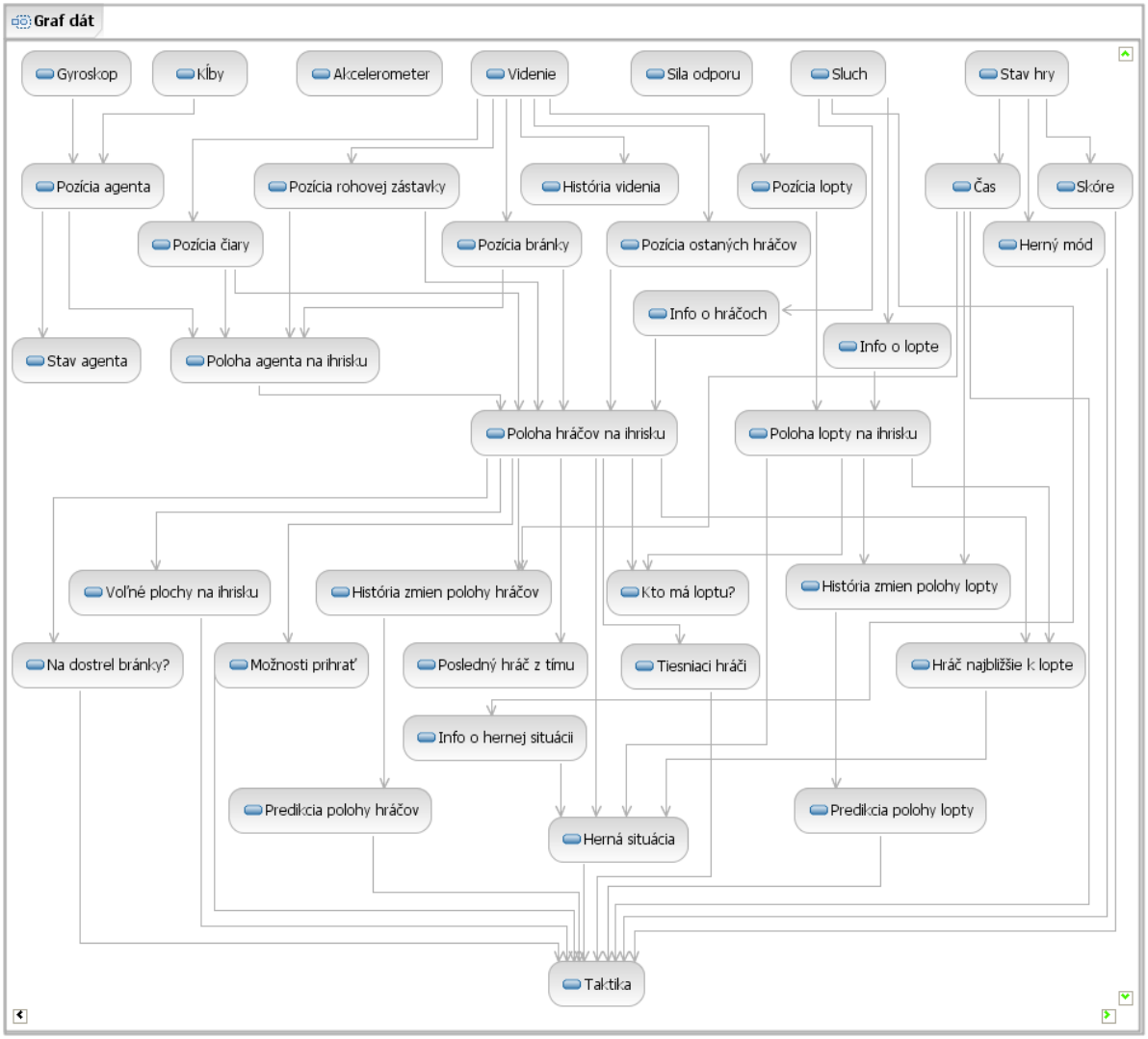
Obrázok 1: Metóda pre vypočítanie vzdialenosti medzi loptou a hráčom

Graf dát

Graf dát predstavuje informácie, ktoré dokáže agent o svete získať. Informácie sú na rôznych stupňoch abstrakcie. Najnižšiu vrstvu predstavujú tie, ktoré prichádzajú zo servera a predstavujú dáta, ktoré agent dostáva zo snímačov. V grafe sú to uzly, do ktorých nevchádza žiadna hrana. Model grafu dát je zobrazený na obrázku 1.

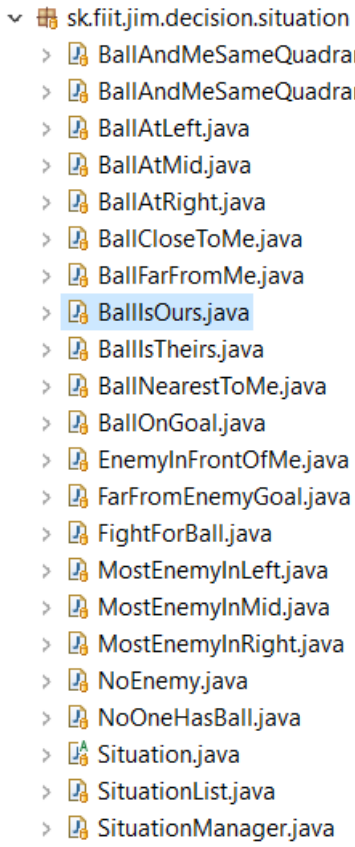
V grafe môžeme vidieť ako prebieha videnie lopty hráčom. Od počiatočného videnia, kde získa Agent pozíciu lopty, následne určí polohu lopty na ihrisku. A podľa tejto polohy, ďalej vyhodnotí rôzne iné situácie ako, kto má loptu? (druhý tím, náš tím), ako ďaleko je daný Agent od lopty, či je pri nej najbližšie a podobne. Následne podľa týchto informácií určí hernú situáciu a podľa nej taktiku, ktorá sa vykoná.

Okrem toho berie do úvahy aj históriu zmien polohy lopty v závislosti od času, ktorá mu slúži pri predikcii polohy lopty.



Obrázok 2: Graf dát

Situácie v závislosti od polohy lopty



Obrázok 3: Situácie, ktoré môžu nastať

```java
public class NoOneHasBall extends Situation {

    private AgentModel agentModel = AgentModel.getInstance();

    /**
     * Situations when no team has ball posession
     * @return boolean - true if no team has ball posession
     */
    @Override
    public boolean checkSituation() {
        if (this.agentModel.getDistanceNereastToBall(WorldModel.getInstance()
                .getTeamPlayers()) >= Situation.EDGE_DISTANCE_FROM_BALL
                && this.agentModel.getDistanceNereastToBall(WorldModel
                .getInstance().getOpponentPlayers()) >= Situation.EDGE_DISTANCE_FROM_BALL
                && agentModel.getDistanceFromBall() >= Situation.EDGE_DISTANCE_FROM_BALL) {
            return true;
        }
        return false;
    }
}
```

Obrázok 4: Ukážka kódu pre skontrolovanie situácie, kedy nikto nemá loptu