**Popis celkového konceptu fungovania agenta**

Agent na ihrisku je riadený kódom, ktorý sa skladá z viacerých modulov navzájom prepojených. Každý modul má inú funkcionalitu. Moduly sú v hierarchií.

# Plánovač

Na vrchu hierarchie je plánovač, ktorý riadi celkové správanie sa agenta. Plánovačov môže byť niekoľko, aktivujú sa na základe stavu hry a taktiky. Plánovač vytvára inštancie vyšších pohybov a aktivuje ich na základe stavu hry a hráča.

Hlavný cyklus beží po inicializácii a načítaní agenta. Hlavný cyklus obsahuje v sebe aj pustenie plánovača.

Plánovač je ruby script, ktorý je uložený v **./scripts/plan/**.

Defaultný plánovač je nastavený v skripte **settings.rb** a volá sa príkazom *Settings.setValue("Planner", "Nazov\_Triedy\_Planovaca")*.

# Vyššie pohyby – high skills

Vyššie pohyby sa taktiež rozhodujú na základe stavov agenta. V tomto prípade ale ide o špecifické stavy vzhľadom na daný high-skill. Napríklad v prípade high-skillu *„Choď k lopte“* sa vyhodnocuje stav agenta voči lopte. Tieto stavy sa vyhodnotia niekoľko krát počas vykonávania high-skillu. Hlavná funkcia high-skillov je volanie low-sklillov, čo vykonáva práve na základe vyhodnotenia stavu agenta.

High-skills sú ruby skripty, fyzicky uložené v **./Jim/scripts/high\_skills**

## Vykonávanie high-skillov

V high-skilloch použité nižšie pohyby musia mať aspoň jednu fázu, ktorá je označená ako konečná fáza, t.j. fázu v ktorej low-skill končí.

High-skill na základe stavu agenta sa rozhoduje, ktorý low-skill bude zavolaný. Low-skill volá funkciou *pickLowSkill*. V prípade zistenia stavu, ktorý v danom high-skill nie je ošetrený, high-skill končí a vráti NULL pre plánovač. Plánovač sa následne rozhodne ako ošetriť vzniknutý stav.

# Nižšie pohyby – low-skills

Nižšie pohyby sú vytvorené jednoduchou štruktúrou. Sú poskladané z radov natáčaní kĺbov NAO robota združené do stavov a fáz.

Low-skills sú fyzicky XML súbory uchovávané v **./Jim/moves**

# Diagram tried pohybov

XML súbory nižších pohybov a ruby skripty vyšších pohybov figurujú v zdrojovom kóde nasledovne:

Niektoré z tried existujú aj v Jave a rovnaké alebo podobné triedy sú aj v Ruby skriptoch:

* **LowSkill** (java)- trieda sk.fiit.jim.agent.moves.LowSkill
* **HighSkill** (java) - trieda sk.fiit.jim.agent.skills.HighSkill
* **LowSkill** (ruby) - trieda LowSkill z scripts/high\_skills/low\_skill.rb, ktorá dedí triedu Ruby-HighSkill
* **HighSkill** (ruby) - trieda RubyHighSkill z scripts/high\_skills/ruby\_high\_skill.rb, ktorá dedí triedu Java-HighSkill



# Stavy agenta

Stavy agenta sa určujú pomocou údajov z Modela sveta. Určuje sa absolútna poloha agenta na ihrisku, poloha lopty a ostatných hráčov. Určuje sa relatívna poloha agenta a lopty voči iným objektom na ihrisku. Ďalej sa určuje natočenie agenta atď. Údaje sú získané pomocou receptorov agenta z jednotlivých častí modelu sveta:

* Agent model - obsahuje určovanie pozície hráča a stranu tímu hráča. Ďalej obsahuje metódu na výpočet pohybového vektora. Vracia informácie o tom či agent stojí, alebo je na zemi, o natočení a pozícii.
* Environment model - ukladá statické informácie o stave súčasného sveta okolo agenta, ako herný čas, simulačné časy, režim prehrávania, verziu servera, veľkosť ihriska a lopty.
* World model, dynamicObject - stavy objektov na ihrisku, ako hráči lopta atď. Ukladá ich pozíciu a rýchlosť.

# Zhrnutie spolupráce plánovača a pohybov

Na základe stavu agenta sa plánovač rozhodne, čo má agent robiť, t.j. aký high-skill bude zavolaný. Pustený high-skill vykoná pohyb poskladaný z low-skillov na základe stavu a zmeny stavu agenta.

# Diagram konceptu fungovania

