Otvorená implementácia architektúry Agent-Space

Andrej Lúčny

KAI FMFI UK Bratislava

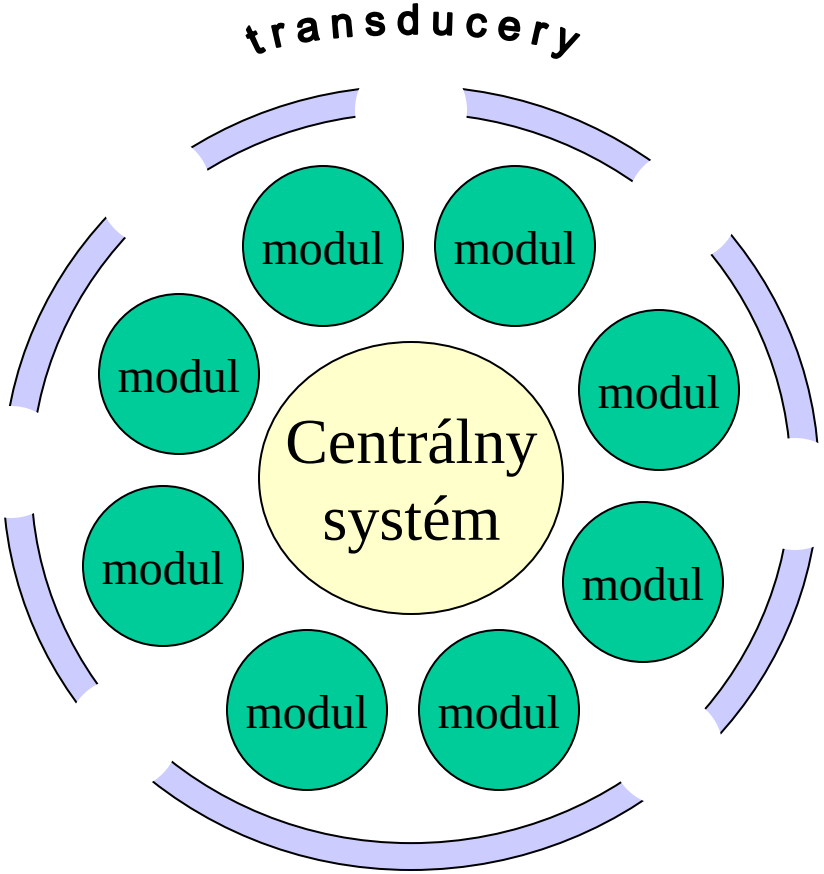
andy@microstep-mis.com

http://www.microstep-mis.com/~andy

Fodorov Model Mysle

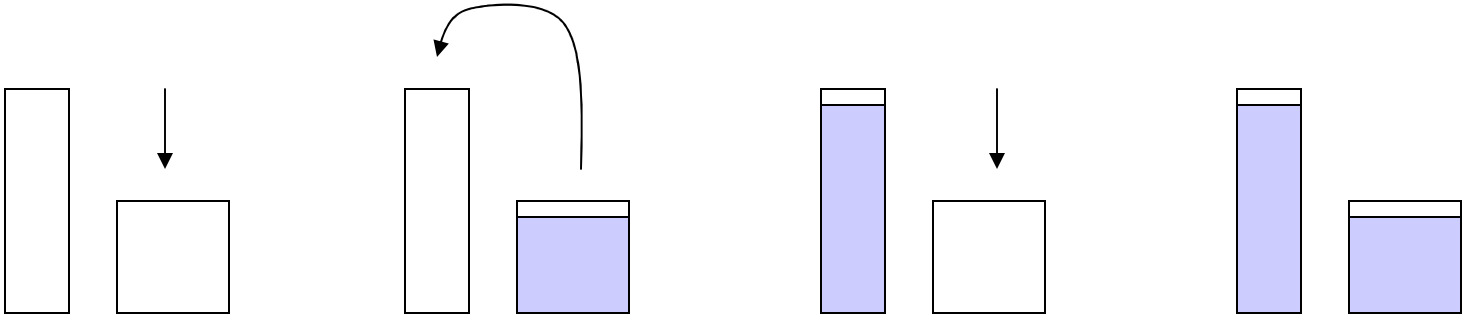
Vlastnosti modulov:

- doménovo špecifické
- anatomicky lokalizovateľné
- mandatórne
- plytký výstup
- informačne uzavreté
- rôzna rýchlosť spracovania
- charakteristické funkcie rozpadu



Piagetove experimenty

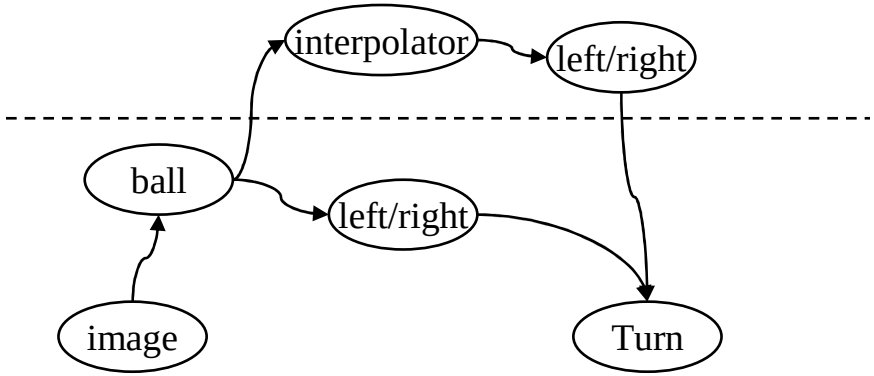
- **V ktorom pohári je viac vody ?**



- **4-ročné a menej si vyberú jeden z pohárov**
- **5-ročné a viac vedia, že vody je rovnako**
- **Ako sa zmení štruktúra v mozgu dieťaťa medzi štvrtým a piatym rokom ?**

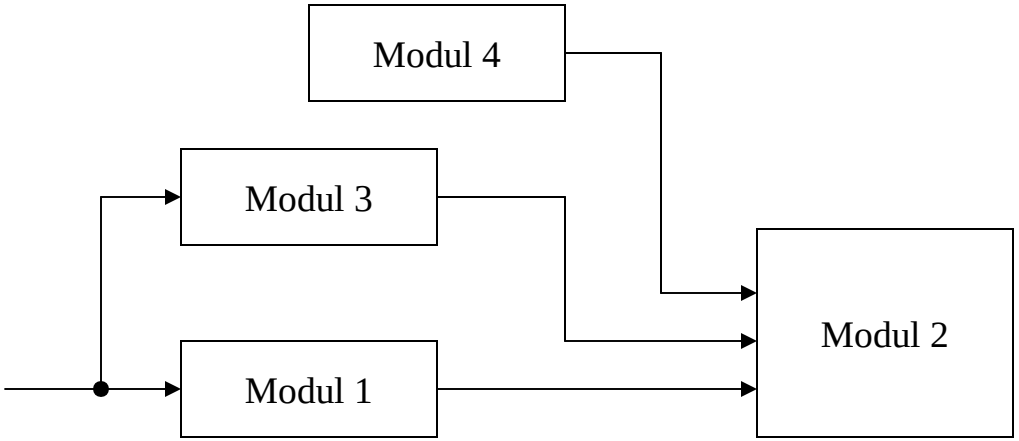
Minského societný model mysle

- Myseľ je distribuovaný systém paralelne bežiacich agentov, ktorí vzájomne komunikujú
- Vonkajšie prejavy systému vyplývajú z lokálnej interakcie medzi agentami
- Správne vonkajšie prejavy sú dané aktivovaním správnej sady agentov v správnom čase



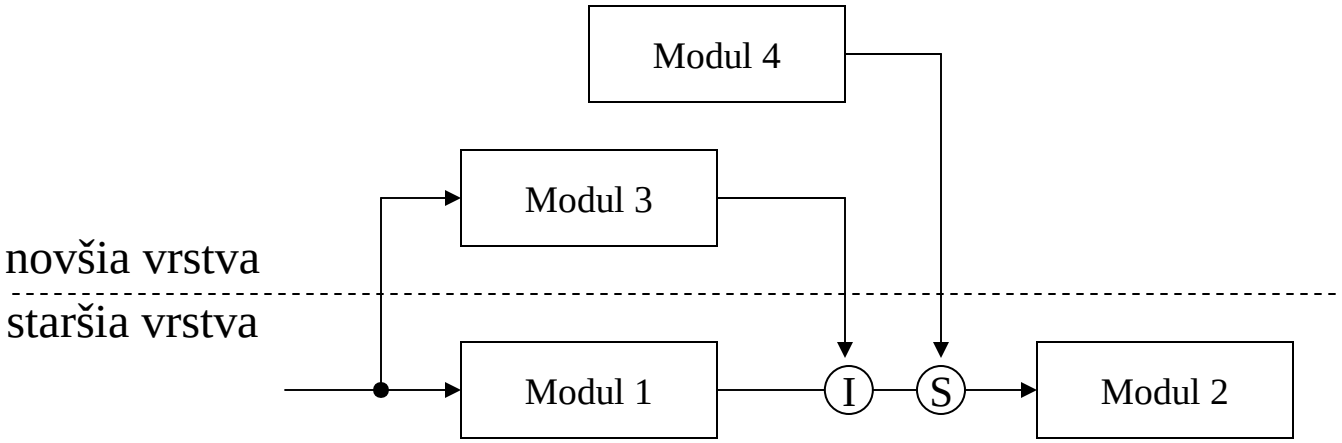
Hardwarová realizácia

- riadenie sa skladá z modulov s pevne definovanými vstupmi a výstupmi
- skladanie systému sa realizuje vhodným prepojením vstupov a výstupov
- problém: rôzna rýchlosť spracovania



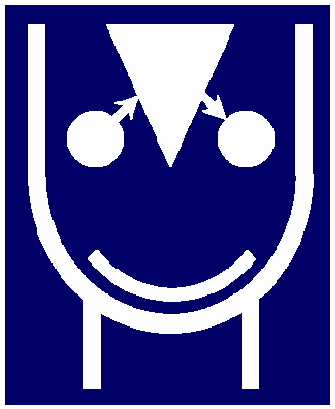
Brooksova subsumpcia

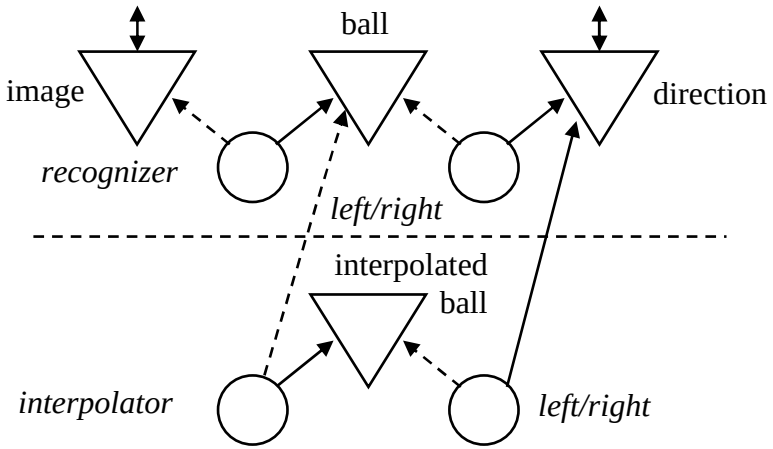
- novšia zahrňuje (subsumuje) staršiu
- novšia vrstva nevyvoláva aktivitu staršej, ale ju reguluje
- odpočúvanie, supresia, inhibícia



Implementácia soc. modelu mysle

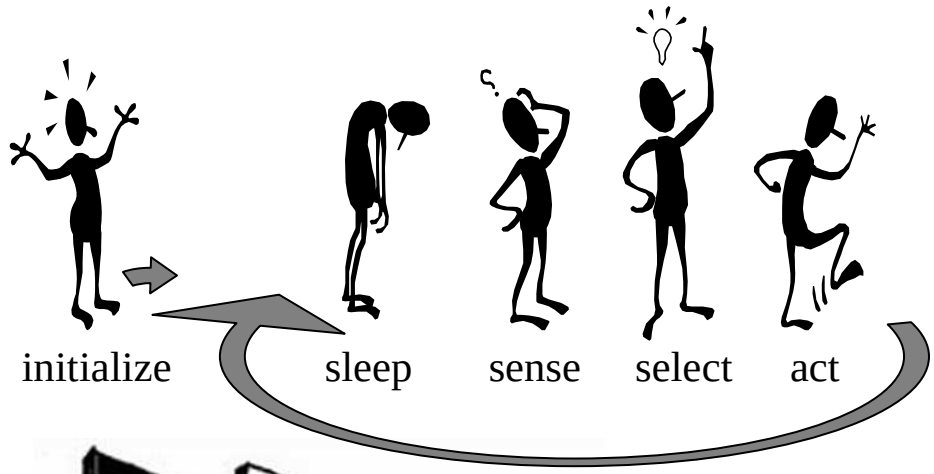
- Architektúra Agent-Space [Lúčny 2004] implementuje modulárny charakter soc. modelu
- www.agentspace.org



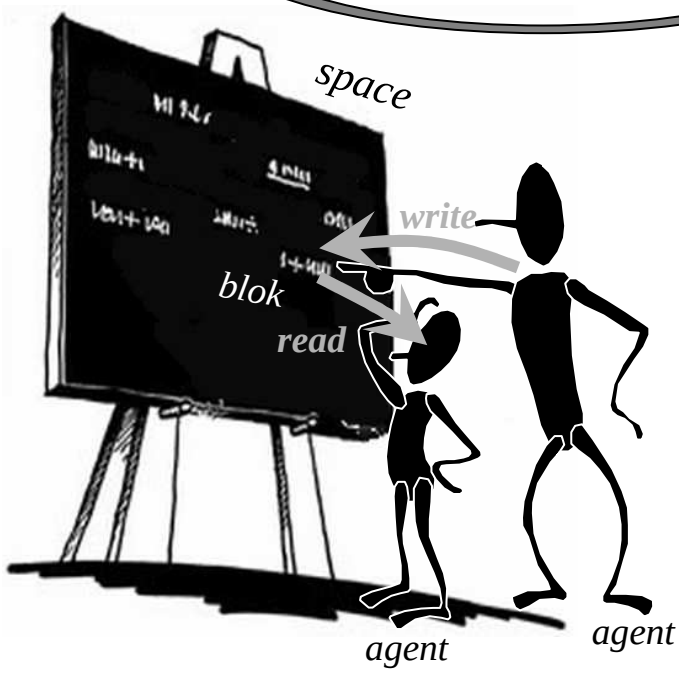


Architektúra Agent-Space

- Systém sa skladá z agentov

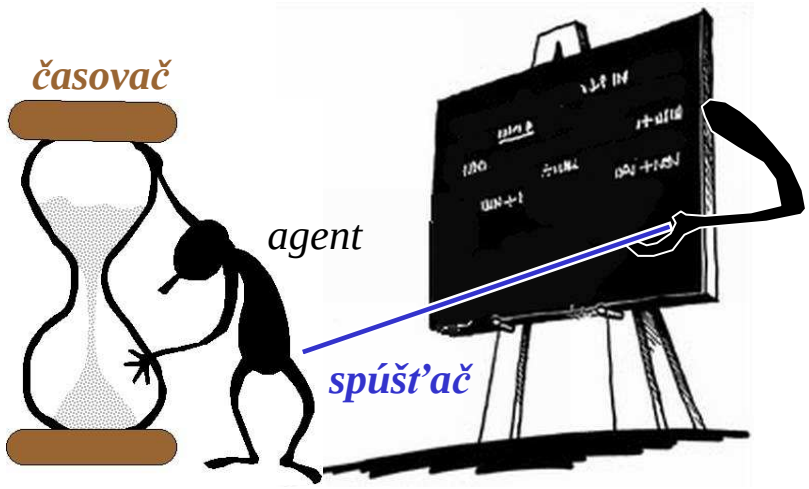


- Agenti medzi sebou komunikujú nepriamo cez Space (čiernu tabuľu)



Implementácia v C++

- Multivláknové prostredie pthreads
- Každý agent má vlastné vlákno
- Može volať metódy singletonu Space
- vlákno agenta je blokované na časovač alebo spúšťač

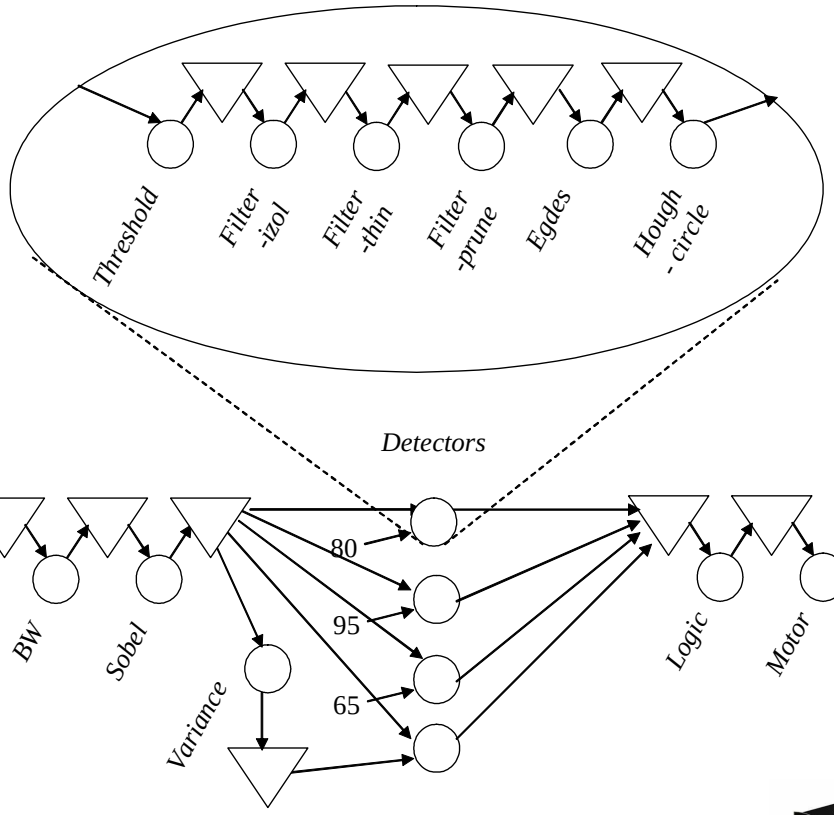


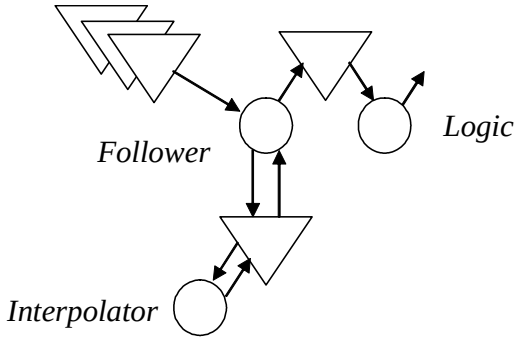
Príklad použitia

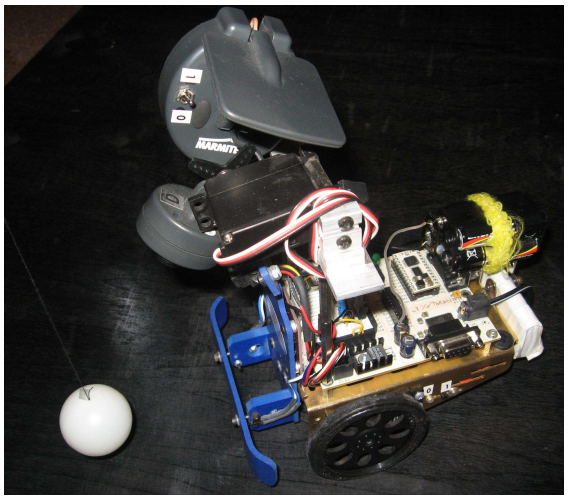
```
#include <iostream>
#include <conio.h>
#include "agentspace.h"
using namespace std;

class MyAgent1 : public Agent {
  private:
    int i;
  protected:

    void init (string args) {
      i = 0;
      timer_attach(1000,1000);
    }

    void sense_select_act (int pid) {
      i++;
      cout << "a := " << i << endl;
      space_write("a",i,1500);
    }

  public:
    MyAgent1 (string args) :
      Agent(args) {};
};
```

```
class MyAgent2 : public Agent {
  protected:

    void init (string args) {
    trigger_attach("*",TRIGGER_MATCHING);
    }

    void sense_select_act (int pid) {
      int a = space_read("a",0);
      cout << "a = " << it->value << endl;
    }

  public:
    MyAgent2 (string args) :
      Agent(args) {};
};

int main () {
        MyAgent1 a1("");
        MyAgent2 a2("");
        getch();
}
```

Príklad použitia:
Riadiaci systém robota

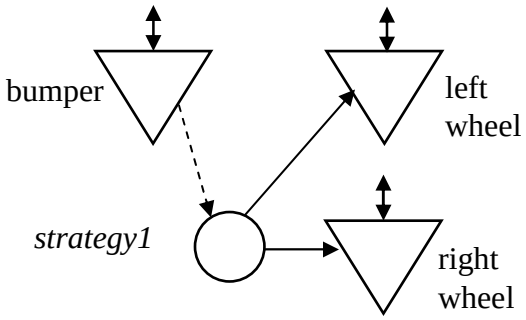


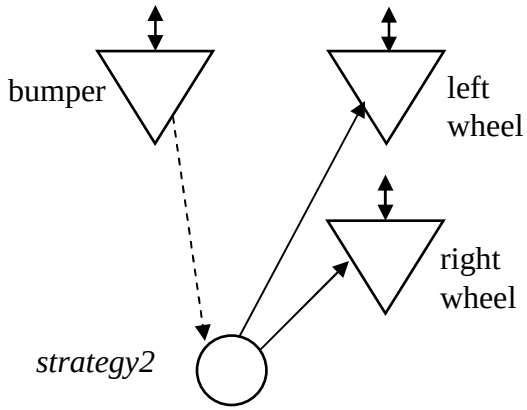


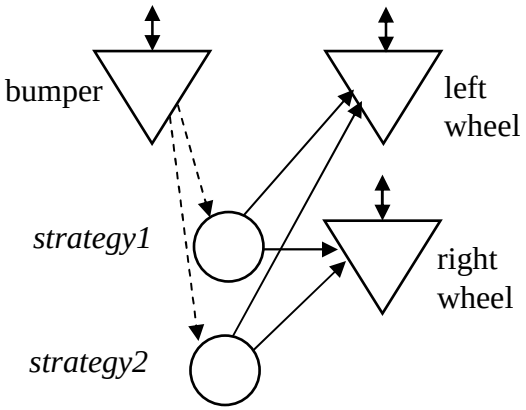


Príklad použitia: kombinácia správaní

- Agent-Space umožnuje bez zábran pridávať a odoberať agentov, i takých ktorí ovládajú tie isté aktuátory robota.







Ďakujem za pozornosť !

www.agentspace.org

Otvorená implementácia
architektúry Agent-Space

Andrej Lúčny
KAI FMFI UK, Bratislava
andy@microstep-mis.com
http://www.microstep-mis.com/~andy