Prog1, C++ befejezés

Magasszintű programozási nyelvek 1 mérnök informatikus BSc előadás

Dr. Bátfai Norbert
egyetemi adjunktus
http://www.inf.unideb.hu/~nbatfai/
Debreceni Egyetem, Informatikai Kar,
Információ Technológia Tanszék
batfai.norbert@inf.unideb.hu
Skype: batfai.norbert

Prog1_7.ppt, v.: 0.0.5, 2012. 04. 24.
http://www.inf.unideb.hu/~nbatfai/#p1

Az óra blogja: http://progpater.blog.hu/
A Nokia Ovi store-ban is elérhető: http://store.ovi.com/content/100794

Felhasználási engedély

Bátfai Norbert
Debreceni Egyetem, Informatikai Kar, Információ Technológia Tanszék
<nbatfai@inf.unideb.hu, nbatfai gmail com>

Célok és tartalom

Előadás

a) Szoftvertervezés
b) Generikus programozás, sablonosztályok
c) Kivételkezelés
d) Kódolási konvenciók

Labor

a) Ismétlés: jelkezelés, setjmp, longjmp
b) Ismétlés: OpenMP
c) Qt slot-signal mechanizmus
d) CORBA (C++, Java, Python visszhang)
e) Robotfoci

Laborkártyák

a) Példás kártyák

Otthoni opcionális feladat

a) A japán világbajnok HELIOS csapat szoftvereinek otthoni tanulmányozása.

Kapcsoldó videók, videómagyarázatok és blogok

http://progpater.blog.hu/2011/04/10/a_kilencedik_tizedik_labor
http://progpater.blog.hu/2011/04/11/imadni_fogjak_a_c_t_egy_emberkent_tiszta_szivbol_3
http://progpater.blog.hu/2011/04/14/egyutt_tamadjuk_meg
http://progpater.blog.hu/2011/04/03/a_nyolcadik_kilencedik_labor
http://progpater.blog.hu/2011/03/27/a_parhuzamossag_gyonyorkodtet
http://progpater.blog.hu/2011/04/23/corba_telefonkonyv

Az írásbeli és a szóbeli vizsgán bármi (jegyzet, könyv, forráskód, számítógép mobiltelefon stb.) használható! (Az írásbeli vizsgán beszélni viszont tilos.) Hiszen az én feladatom az lesz, hogy eldöntsem, jól felkészült programozóval, vagy mennyire felkészült programozóval állok szemben.

Minimális gyakorlati cél

1) A hallgató tudjon egyszerű osztálysablonokat (pl. a tárgyalt Verem vagy BinFa) létrehozni.
2) Qt kódolási konvenciók
3) Robotfoci a 2. védés!!!

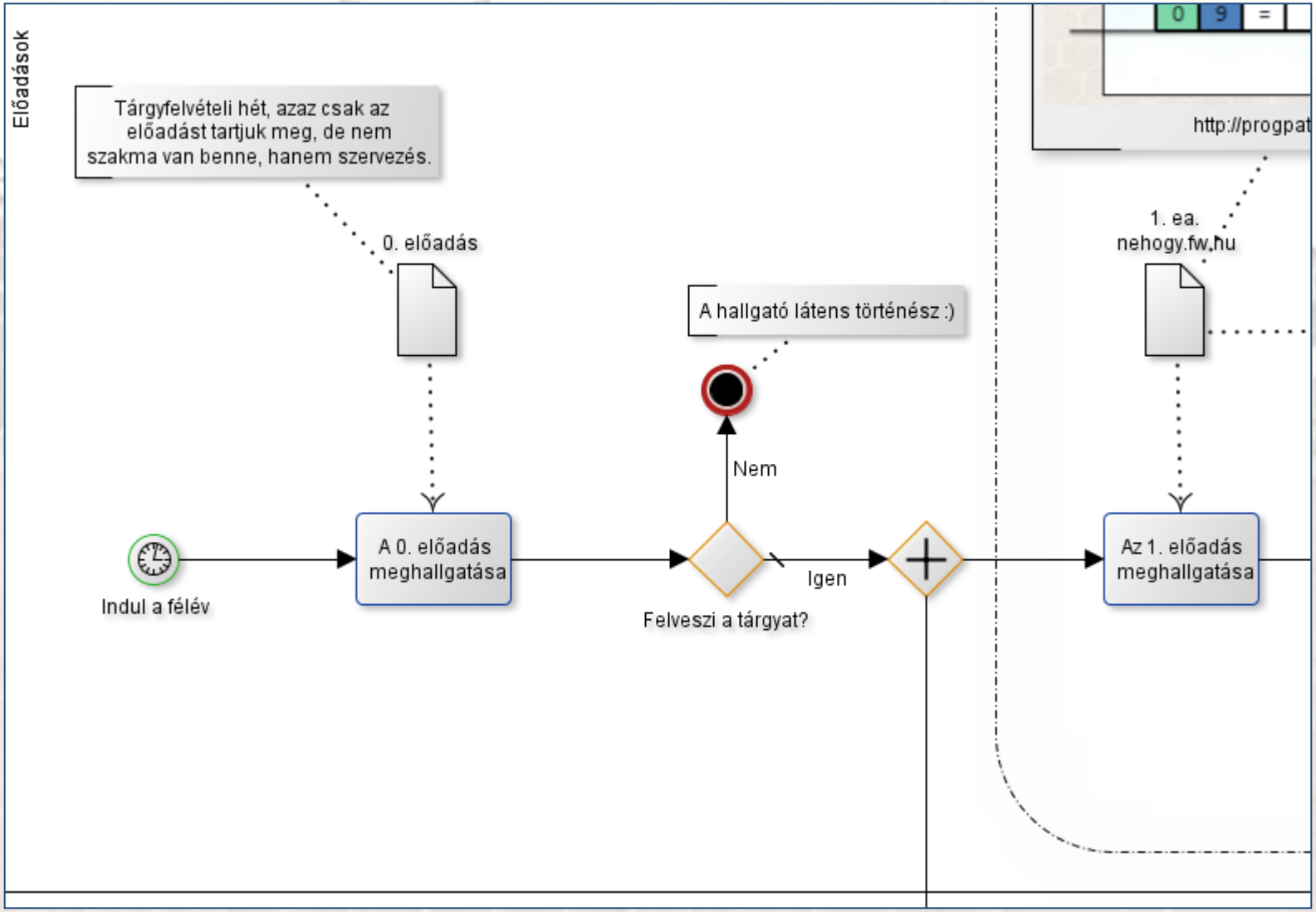
Az svn tároló elérése kapcsán:
http://progpater.blog.hu/2011/02/05/az_elso_labor/fullcommentlist/1#c12856691

Minimális elméleti cél

1) OPC, LSP
2) Korai, késői kötés
3) Sablonok

Kurzustérkép

BPMN (Business Process Modeling Notation)



http://www.inf.unideb.hu/~nbatfai/prog1_terkep/

Szoftverfejlesztés

Open-Closed elv (OCP)
(kiterjeszt, nem módosít) ha sérül az LSP, sérül az OCP is

Ismétlés: Liskov féle helyettesítési elv

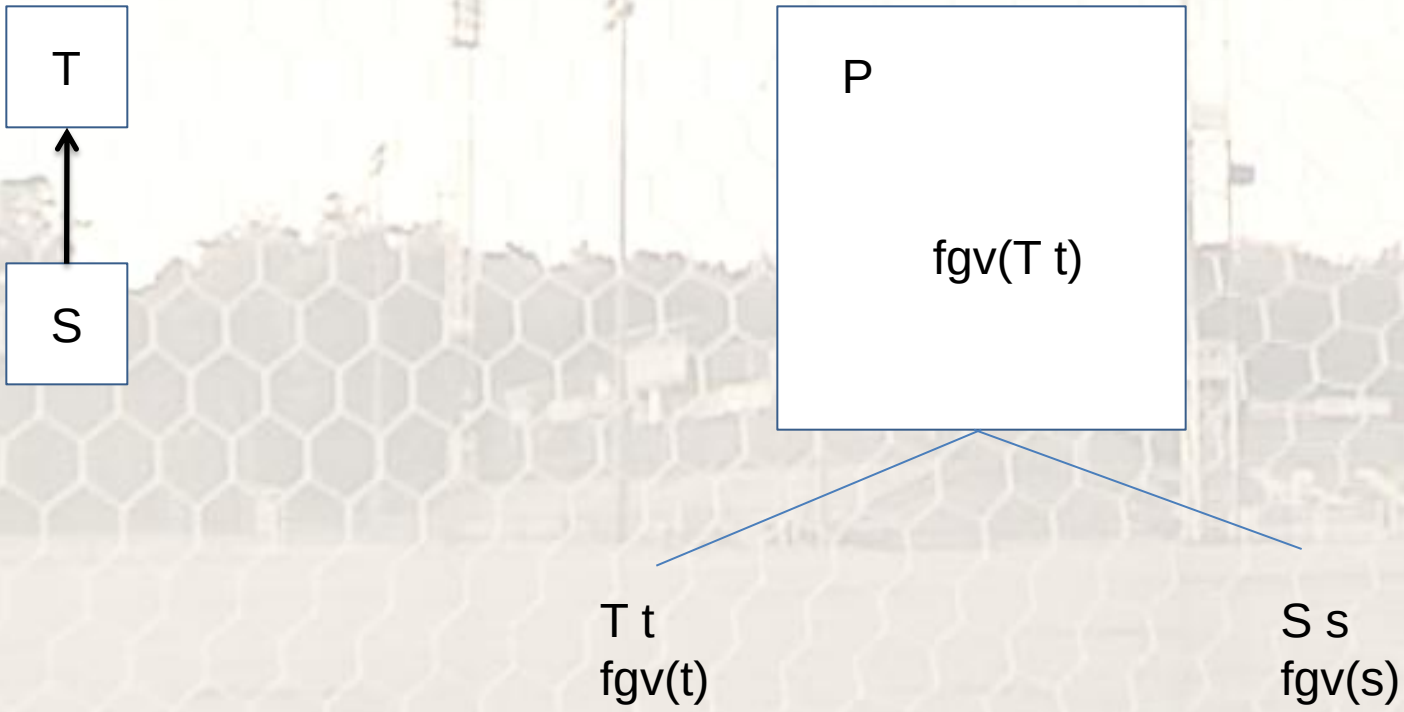
Barbara Liskov: Aata Abstraction and Hierarchy, OOPSLA '87 Addendum to the proceedings on Object-oriented programming systems, languages and applications (Addendum) ACM New York, NY, USA, 1987.
http://portal.acm.org/citation.cfm?doid=62138.62141

Liskov Substitution Principle (LSP)

3.3. Type Hierarchy

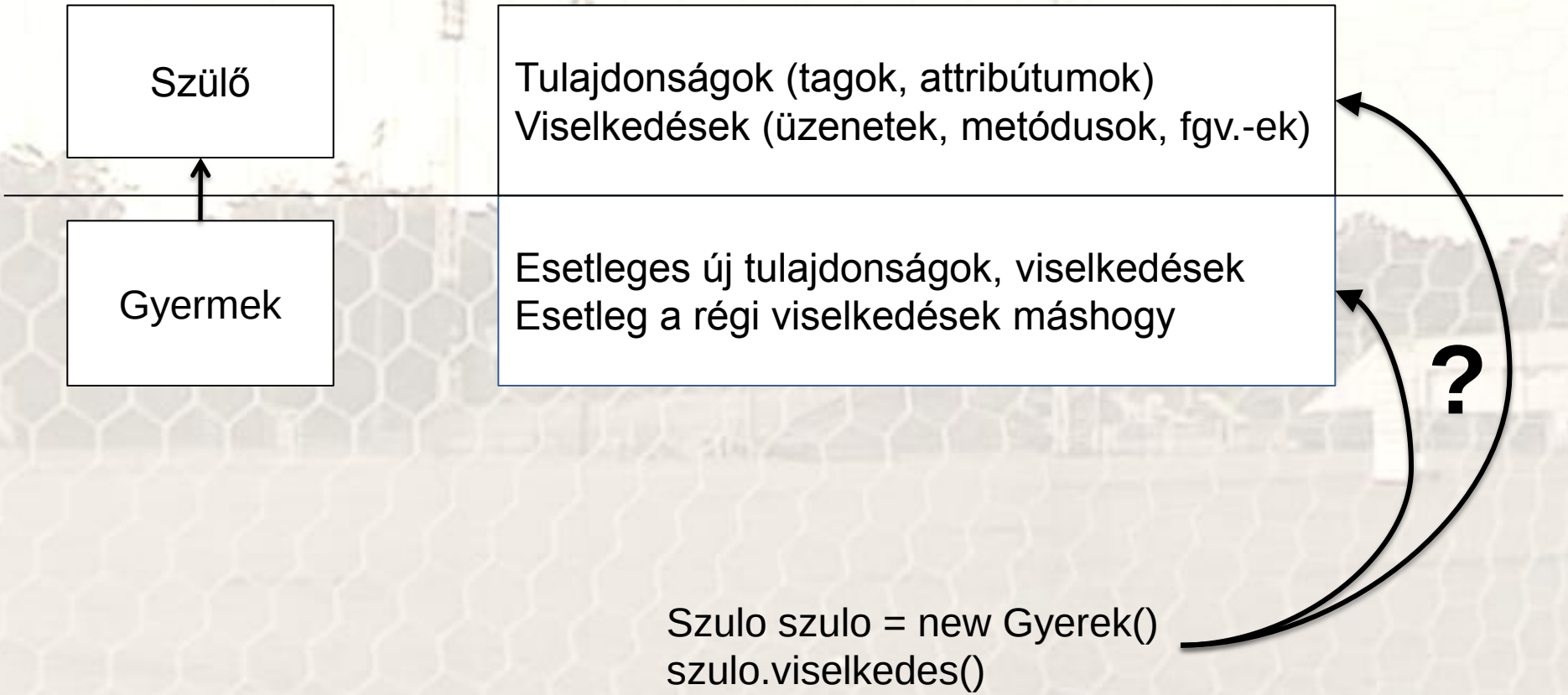
A type hierarchy is composed of subtypes and supertypes. The intuitive idea of a *subtype* is one whose objects provide all the behavior of objects of another type (the *supertype*) plus something extra. What is wanted here is something like the following substitution property [6]: If for each object o_1 of type S there is an object o_2 of type T such that for all programs P defined in terms of T, the behavior of P is unchanged when o_1 is substituted for o_2, then S is a subtype of T. (See also [2, 17] for other work in this area.)

Összefoglalás: Liskov féle helyettesítési elv

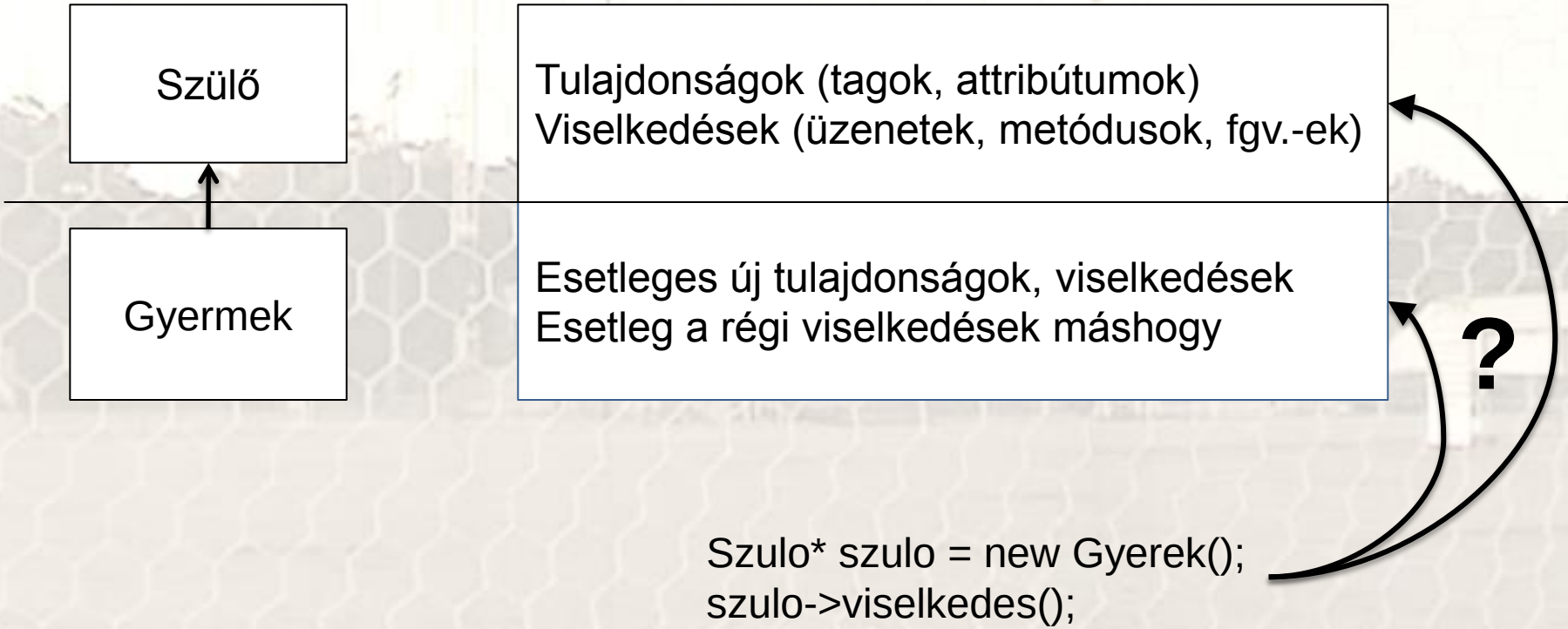


(Az ősökkel működő függvény működjön a gyermekekkel is, ugyanúgy!)

Polimorfizmus



Dinamikus vagy késői kötés




```
Szulo* szulo = new Gyerek();
szulo->viselkedes();
```

Ha a válasz futási (és nem fordítási) időben dől el.

Dinamikus vagy késői kötés

Szulo* szulo = new Gyerek();

Statikus (deklarációs) típus

Dinamikus (példányosított) típus

Virtuális fgv.-ek hívása a dinamikus típus alapján történik.

Java

Szulo szulo = new Gyerek()

Minden objektum referencia.
Mindig dinamikus a kötés.

De ezzel nem küldhetjük a
Gyerek által hozott új üzeneteket

Szulo szulo = new Gyerek()

C++

Lehet:
Szulo& szulo … referencia
Szulo* szulo … mutató
Szulo szulo … objektum

Csak akkor van dinamikus kötés, ha a viselkedés virtuálisra (virtual kulcsszó az ősben) van deklarálva.

Ugyanúgy igaz, hogy ösosztály referencián vagy pointeren keresztül, csak az ős üzenetei küldhetőek.

Ismétlés

```
K:Param 42
K:Szulo 0x7fff1600ed30
Szulo::tulterhelt
Szulo::tulterhelt
K:Param 42
K:Szulo 0x7fff1600ed10
K:Param2 42
K:Gyermek 0x7fff1600ed10
Gyermek::tulterhelt
Szulo::tulterhelt
K:Param 42
K:Szulo 0x152d030
K:Param2 42
K:Gyermek 0x152d030
Szulo::tulterhelt
Szulo::tulterhelt
D:Szulo
D:Param
D:Gyermek
D:Param2
D:Szulo
D:Param
D:Szulo
D:Param
```

```
void fgv(Szulo & szulo) {

  szulo.tulterhelt();

}

int
main (int argc, char *argv[])
{
  Szulo sz;
  sz.tulterhelt();
  fgv(sz);

  Gyermek gy;
  gy.tulterhelt();
  fgv(gy);

  Szulo * szp = new Gyermek();
  szp->tulterhelt();
  fgv(*szp);

  delete szp;

  return 0;
}
```

polimorf mutató

```
K:Param 42
K:Szulo 0x7fff40f38fb8
Szulo::tulterhelt
Szulo::tulterhelt
K:Param 42
K:Szulo 0x7fff40f38f88
K:Param2 42
K:Gyermek 0x7fff40f38f88
Gyermek::tulterhelt
Gyermek::tulterhelt
K:Param 42
K:Szulo 0x1c7b038
K:Param2 42
K:Gyermek 0x1c7b038
Gyermek::tulterhelt
Gyermek::tulterhelt
D:Szulo
D:Param
D:Gyermek
D:Param2
D:Szulo
D:Param
D:Szulo
D:Param
```

Ismétlés: a saját Verem osztályunk

```
class Verem
{

public:
Verem (int m = 1024):meret (m), verem (new char[m])
  {
    sp = -1;
  }
  Verem (Verem& v):meret (v.meret), verem (new char[v.meret])
  {
    sp = v.sp;

    for (int i = 0; i < v.meret; ++i)
      verem[i] = v.verem[i];
  }
  ~Verem ()
  {
    delete[]verem;
  }
  Verem& operator=(Verem& v)
  {
    char *ujverem = new char[v.meret];
    sp = v.sp;
    meret = v.meret;

    for (int i = 0; i < v.meret; ++i)
      ujverem[i] = v.verem[i];

    delete [] verem;
    verem = ujverem;

    return *this;
  }
  char pop ()
  {
    return verem[sp--];
  }
  void push (char c)
  {
    verem[++sp] = c;
  }
  int getMeret ()
  {
    return meret;
  }
  int getDarab ()
  {
    return sp + 1;
  }

private:
  char *verem;
  int meret;
  int sp;
};
```

Verem
- verem : char* - meret : int - sp : int
+ Verem(m : int) + Verem(v : Verem&) + ~ Verem() + operator =(v : Verem&) : Verem& + pop() : char + push(c : char) + getMeret() : int + getDarab() : int

Dupla lebegőpontosok verme

```cpp
#include <iostream>

class Verem
{
    int meret;
    double *verem;
    int sp;

public:
    Verem (int m = 1024):meret (m), verem (new double[m])
    {
        sp = -1;
    }
    Verem (Verem& v):meret (v.meret), verem (new double[v.meret])
    {
        sp = v.sp;

        for (int i = 0; i < v.meret; ++i)
            verem[i] = v.verem[i];
    }
    ~Verem ()
    {
        delete[]verem;
    }
    Verem& operator=(Verem& v)
    {
        double *ujverem = new double[v.meret];
        sp = v.sp;
        meret = v.meret;

        for (int i = 0; i < v.meret; ++i)
            ujverem[i] = v.verem[i];

        delete [] verem;
        verem = ujverem;

        return *this;
    }
    double& operator[](int i) {
        return verem[i];
    }
    double pop ()
```

```cpp
int
main (int argc, char
{
  Verem v (512);

  v.push (5.5);
  v.push (6.5);

  Verem u;

  u.push (7.5);
  u.push (8.5);

  u = v;

  while (u.getDarab ())
    std::cout << "u: " << u.pop () << std::endl;

  while (v.getDarab ())
    std::cout << "v: " << v.pop () << std::endl;

  return 0;
}
```

Verem
- meret : int - verem : double* - sp : int
+ Verem(m : int) + Verem(v : Verem&) + ~ Verem() + operator =(v : Verem&) : Verem& + operator [ ](i : int) : double& + pop() : double + push(c : double) + getMeret() : int + getDarab() : int

Logikaiak verme

```
#include <iostream>

class Verem
{
    int meret;
    bool *verem;
    int sp;

public:
    Verem (int m = 1024):meret (m), verem (new bool[m])
    {
        sp = -1;
    }
    Verem (Verem& v):meret (v.meret), verem (new bool[v.meret])
    {
        sp = v.sp;

        for (int i = 0; i < v.meret; ++i)
            verem[i] = v.verem[i];
    }
    ~Verem ()
    {
        delete[]verem;
    }
    Verem& operator=(Verem& v)
    {
        bool *ujverem = new bool[v.meret];
        sp = v.sp;
        meret = v.meret;

        for (int i = 0; i < v.meret; ++i)
            ujverem[i] = v.verem[i];

        delete [] verem;
        verem = ujverem;

        return *this;
    }
    bool& operator[](int i) {
        return verem[i];
    }
    bool pop ()
```

```
int
main (int argc, char *argv[])
{
  Verem v (512);

  v.push (true);
  v.push (true);

  Verem u;

  u.push (false);
  u.push (false);

  u = v;

  while (u.getDarab ())
    std::cout << "u: " << u.pop () << std::endl;

  while (v.getDarab ())
    std::cout << "v: " << v.pop () << std::endl;

  return 0;
}
```

Sztingek verme

```
#include <iostream>

class Verem
{
    int meret;
    std::string *verem;
    int sp;

public:
    Verem (int m = 1024):meret (m), verem (new std::string[m])
    {
        sp = -1;
    }
    Verem (Verem& v):meret (v.meret), verem (new std::string[v.meret])
    {
        sp = v.sp;

        for (int i = 0; i < v.meret; ++i)
            verem[i] = v.verem[i];
    }
    ~Verem ()
    {
        delete[]verem;
    }
    Verem& operator=(Verem& v)
    {
        std::string *ujverem = new std::string[v.meret];
        sp = v.sp;
        meret = v.meret;

        for (int i = 0; i < v.meret; ++i)
            ujverem[i] = v.verem[i];

        delete [] verem;
        verem = ujverem;

        return *this;
    }
    std::string& operator[](int i) {
        return verem[i];
    }
    std::string pop ()
```

```
int
main (int argc, char *argv[])
{
  Verem v (512);

  v.push ("alma");
  v.push ("korte");

  Verem u;

  u.push ("dio");
  u.push ("banan");

  u = v;

  while (u.getDarab ())
    std::cout << "u: " << u.pop () << std::endl;

  while (v.getDarab ())
    std::cout << "v: " << v.pop () << std::endl;

  return 0;
}
```

Osztálysablon definíció

```
class BinFa
{

};
```

Sablonparaméter lista

```
template <class CsomopontTartalomTipus> class BinFa
{

};
```

Osztálysablon definíció, logikaiak verme

```
#include <iostream>

template <class Tipus> class Verem
{
    int meret;
    Tipus *verem;
    int sp;

public:
    Verem (int m = 1024):meret (m), verem (new Tipus[m])
    {
        sp = -1;
    }
    Verem (Verem& v):meret (v.meret), verem (new Tipus[v.meret])
    {
        sp = v.sp;

        for (int i = 0; i < v.meret; ++i)
            verem[i] = v.verem[i];
    }
    ~Verem ()
    {
        delete[]verem;
    }
    Verem& operator=(Verem& v)
    {
        Tipus *ujverem = new Tipus[v.meret];
        sp = v.sp;
        meret = v.meret;

        for (int i = 0; i < v.meret; ++i)
            ujverem[i] = v.verem[i];

        delete [] verem;
        verem = ujverem;

        return *this;
    }
    Tipus& operator[](int i) {
        return verem[i];
    }
    Tipus pop ()
```

```
int
main (int argc, char *argv[])
{
  Verem<bool> v (512);

  v.push (true);
  v.push (true);

  Verem<bool> u;

  u.push (false);
  u.push (false);

  u = v;

  while (u.getDarab ())
    std::cout << "u: " << u.pop () << std::endl;

  while (v.getDarab ())
    std::cout << "v: " << v.pop () << std::endl;

  return 0;
}
```

Osztálysablon definíció, egész verem

```
#include <iostream>

template <class Tipus> class Verem
{
    int meret;
    Tipus *verem;
    int sp;

public:
    Verem (int m = 1024):meret (m), verem (new Tipus[m])
    {
        sp = -1;
    }
    Verem (Verem& v):meret (v.meret), verem (new Tipus[v.meret])
    {
        sp = v.sp;

        for (int i = 0; i < v.meret; ++i)
            verem[i] = v.verem[i];
    }
    ~Verem ()
    {
        delete[]verem;
    }
    Verem& operator=(Verem& v)
    {
        Tipus *ujverem = new Tipus[v.meret];
        sp = v.sp;
        meret = v.meret;

        for (int i = 0; i < v.meret; ++i)
            ujverem[i] = v.verem[i];

        delete [] verem;
        verem = ujverem;

        return *this;
    }
    Tipus& operator[](int i) {
        return verem[i];
    }
    Tipus pop ()
```

```
int
main (int argc, char *argv[])
{
  Verem<int> v (512);

  v.push (55);
  v.push (56);

  Verem<int> u;

  u.push (65);
  u.push (66);

  u = v;

  while (u.getDarab ())
    std::cout << "u: " << u.pop () << std::endl;

  while (v.getDarab ())
    std::cout << "v: " << v.pop () << std::endl;

  return 0;
}
```

Vermek verme

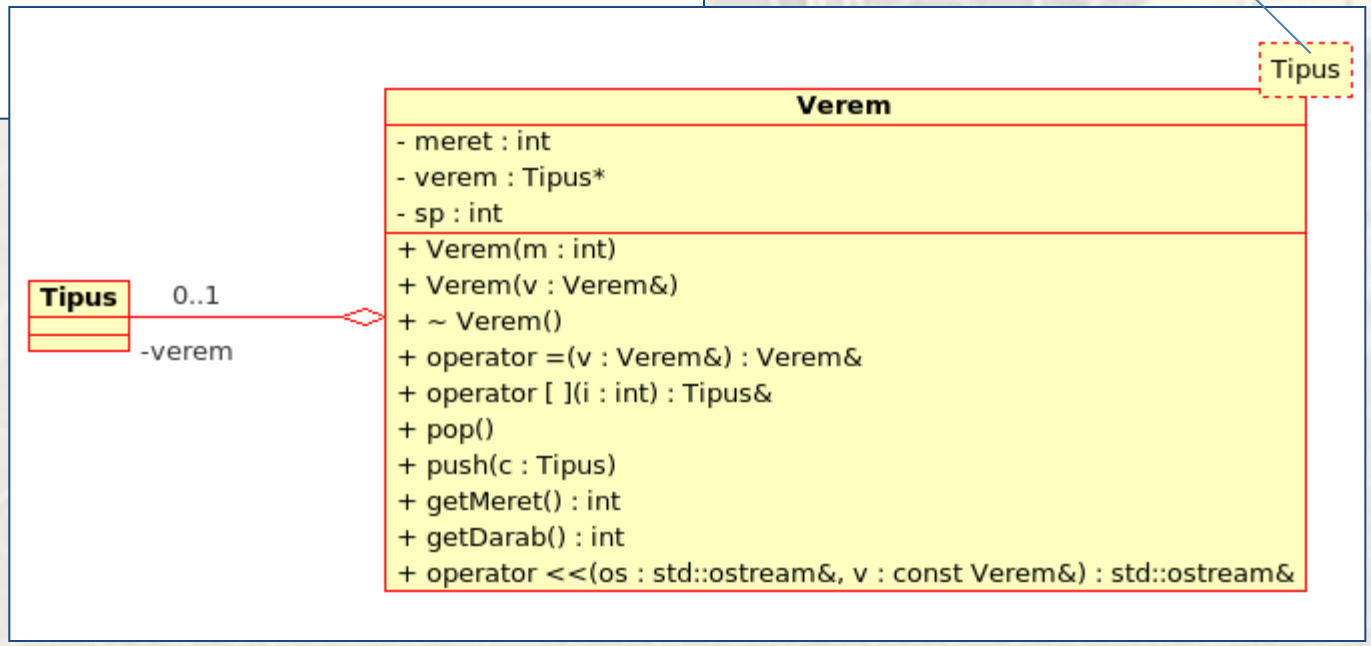
```
#include <iostream>

template <class Tipus> class Verem
{
    int meret;
    Tipus *verem;
    int sp;

public:
    Verem (int m = 1024):meret (m), verem (new Tipus[m])
    {
        sp = -1;
    }
    Verem (Verem& v):meret (v.meret), verem (new Tipus[v.meret])
    {
        sp = v.sp;

        for (int i = 0; i < v.meret; ++i)
            verem[i] = v.verem[i];
    }
    ~Verem ()
    {
        delete[]verem;
    }
    Verem& operator=(Verem& v)
    {
        Tipus *ujverem = new Tipus[v.meret];
        sp = v.sp;
        meret = v.meret;

        for (int i = 0; i < v.meret; ++i)
            ujverem[i] = v.verem[i];

        delete [] verem;
        verem = ujverem;

        return *this;
    }
    Tipus& operator[](int i) {
        return verem[i];
    }
    Tipus pop ()
```

```
int
main (int argc, char *argv[])
{
    Verem<char> v (512), u (128);
    Verem< Verem<char> > vv (32), uu (16);

    v.push ('0');
    v.push ('1');

    u.push ('2');
    u.push ('3');

    vv.push (v);
    vv.push (v);

    uu.push (u);
    uu.push (u);

    std::cout << "u: " << u << std::endl;

    u = v;

    std::cout << "v: " << v << std::endl;
    std::cout << "u: " << u << std::endl;

    std::cout << "uu: " << uu << std::endl;

    uu = vv;

    std::cout << "vv: " << vv << std::endl;
    std::cout << "uu: " << uu << std::endl;

    return 0;
}
```

Vermek verme

```
    Tipus pop () {
        return verem[sp--];
    }
    void push (Tipus c) {
        verem[++sp] = c;
    }
    int getMeret () const {
        return meret;
    }
    int getDarab () const {
        return sp + 1;
    }
    friend std::ostream& operator<< (std::ostream& os, const Verem& v) {

        os << "[ " << v.getMeret() << ", " << v.getDarab() << " | ";
        int mennyi = v.getDarab();
        for (int i=0; i<mennyi; ++i)
            os << v[i] << ", ";
        os << " ]";
        return os;
    }

};
```

sablonparaméterek



Vermek verme

```
[norbi@sgu ~]$ g++ verem14.cpp -o verem
[norbi@sgu ~]$ ./verem
u: [ 128, 2 | 2, 3,  ]
v: [ 512, 2 | 0, 1,  ]
u: [ 512, 2 | 0, 1,  ]
uu: [ 16, 2 | [ 128, 2 | 2, 3,  ], [ 128, 2 | 2, 3,  ],  ]
vv: [ 32, 2 | [ 512, 2 | 0, 1,  ], [ 512, 2 | 0, 1,  ],  ]
uu: [ 32, 2 | [ 512, 2 | 0, 1,  ], [ 512, 2 | 0, 1,  ],  ]
```

```
int
main (int argc, char *argv[])
{
    Verem<char> v (512), u (128);
    Verem< Verem<char> > vv (32), uu (16);

    v.push ('0');
    v.push ('1');

    u.push ('2');
    u.push ('3');

    vv.push (v);
    vv.push (v);

    uu.push (u);
    uu.push (u);

    std::cout << "u: " << u << std::endl;

    u = v;

    std::cout << "v: " << v << std::endl;
    std::cout << "u: " << u << std::endl;

    std::cout << "uu: " << uu << std::endl;

    uu = vv;

    std::cout << "vv: " << vv << std::endl;
    std::cout << "uu: " << uu << std::endl;

    return 0;
}
```

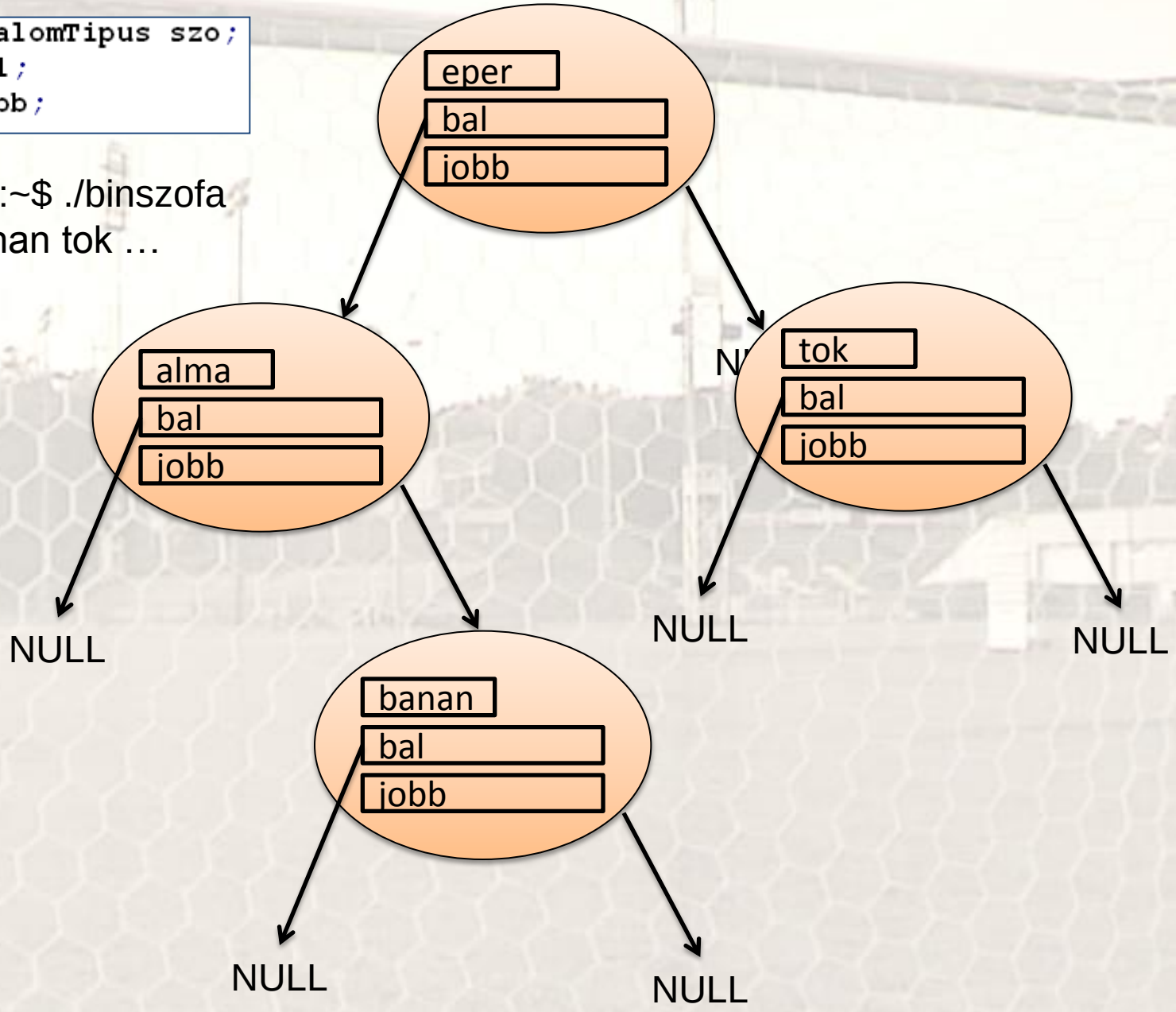
Bináris keresőfák

```
nbatfai@hallg:~$ g++ mains.cpp -o binszofa
nbatfai@hallg:~$ ./binszofa
eper alma banan tok datolya alma piszke barack dinnye dio alma eper
------tok(1, 1)
---------piszke(1, 2)
---eper(2, 0)
------------------dio(1, 5)
--------------dinnye(1, 4)
------------datolya(1, 3)
--------------barack(1, 4)
---------banan(1, 2)
------alma(3, 1)
```

http://progpater.blog.hu/2011/04/11/imadni_fogjak_a_c_t_egy_emberkent_tiszta_szivbol_3#more2816438

```
CsomopontTartalomTipus szo;
Csomopont *bal;
Csomopont *jobb;
```

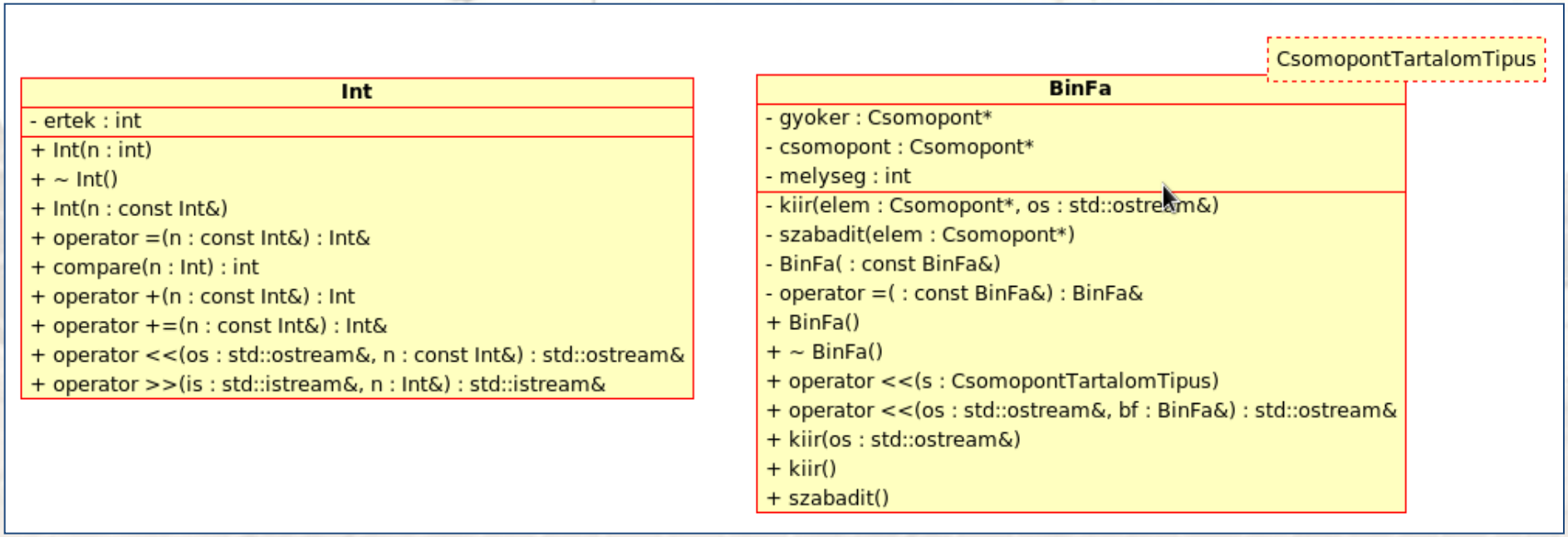
nbatfai@hallg:~$ ./binszofa
eper alma banan tok …



Bináris keresőfák

```
nbatfai@hallg:~$ ./binszofa
alma eper banan tok datolya alma piszke barack dinnye dio alma eper
---------tok(1, 2)
------------piszke(1, 3)
------eper(2, 1)
------------------dio(1, 5)
--------------dinnye(1, 4)
------------datolya(1, 3)
--------------barack(1, 4)
---------banan(1, 2)
---alma(3, 0)
```

Bináris keresőfa sablonosztályos megvalósítása




```
void kiir (void)
    {
        // Sokkal elegánsabb lenne (és más, a bevezetésben nem kibontandó reentráns kérdések miatt is, mert
        // ugye ha most két helyről hívják meg az objektum ilyen függvényeit, tahát ha kétszer kezd futni az
        // objektum kiir() fgv.-e pl., az komoly hiba, mert elromlana a mélység... tehát a mostani megoldásunk
        // nem reentráns) ha nem használnánk a C verzióban globális változókat, a C++ változatban példánytagot a
        // mélység kezelésére: http://progpater.blog.hu/2011/03/05/there_is_no_spoon
        melyseg = 0;
        // ha nem mondta meg a hívó az üzenetben, hogy hova írjuk ki a fát, akkor a
        // sztenderd out-ra nyomjuk
        kiir (&gyoker, std::cout);
    }
```

A bináris keresőfás sablondefiníciónk

```
#ifndef BIN_FA__H
#define BIN_FA__H

#include <iostream>

template <class CsomopontTartalomTipus> class BinFa
{
    class Csomopont
    {
    public:
        Csomopont (CsomopontTartalomTipus s):szo (s), hanyszor (1), bal (0), jobb (0) {};
        ~Csomopont () {};
        Csomopont *balra () {
            return bal;
        }
        Csomopont *jobbra ()
        {
            return jobb;
        }
        void balra (Csomopont * bal)
        {
            this->bal = bal;
        }
```

Tagfüggvények az osztálydefiníción kívül

```
};

template <class CsomopontTartalomTipus>
void BinFa<CsomopontTartalomTipus>::operator<< (CsomopontTartalomTipus s)
{
    int e;

    if (csomopont == NULL)
    {

        csomopont = new Csomopont (s);
        gyoker = csomopont;

    }
    else if ((e = csomopont->tartalma ().compare (s)) == 0)
    {

        csomopont->novel ();

    }
    else if (e > 0)
```

Tagfüggvények az osztálydefiníción kívül

```
};

template <class CsomopontTartalomTipus>
void BinFa<CsomopontTartalomTipus>::operator<< (CsomopontTartalomTipus s)
{
    int e;

    if (csomopont == NULL)
    {

        csomopont = new Csomopont (s);
        gyoker = csomopont;

    }
    else if ((e = csomopont->tartalma ().compare (s)) == 0)
    {

        csomopont->novel ();

    }
    else if (e > 0)
```

```
template <class CsomopontTartalomTipus>
void BinFa<CsomopontTartalomTipus>::kiir (Csomopont * elem, std::ostream& os)
{
```

A sablon példányosítása

```
#include <iostream>
#include "binfa.h"
int
main ()
{
  BinFa < std::string > binSzoFa;
  std::string s;

  while (std::cin >> s)
    {
      binSzoFa << s;
    }

  std::cout << binSzoFa;

  binSzoFa.szabadit ();

  return 0;
}
```

```
int
main () {

    BinFa<Int> binSzamFa;
    Int n;

    while (std::cin >> n)
    {
        binSzamFa << n;
    }

    std::cout << binSzamFa;

    binSzamFa.szabadit ();

    return 0;
}
```

Generikus programozás

Parametrikus polimorfizmus

Kivételkezelés - fogalomkör

a) Kivétel (ha
b) (Tovább)dc
c) Elkapni eg

Korábban: je
figyelmen kív

```
*osztott_memoria_terulet = 42;
*(osztott_memoria_terulet+1) = 0;
sigaction (SIGCHLD, &sa, NULL);
FD_ZERO (&kapu_figyelok);
for (;;)
{
    FD_SET (kapu_figyelo, &kapu_figyelok);
    timeout.tv_sec = 3;
    timeout.tv_usec = 0;
    if ((s = select (kapu_figyelo + 1, &kapu_figyelok, NULL,
                     NULL, &timeout)) == -1)
    {
        switch (errno)
        {
        case EBADF:
            printf ("EBADF\n");
            break;
        case EINTR:
            printf ("EINTR\n");
            break;
        case EINVAL:
            printf ("EINVAL\n");
            break;
        case ENOMEM:
            printf ("ENOMEM\n");
            break;
```

ói))

lsm.: jelek

```
SIGNAL(7)                  Linux Programmer's Manual                  SIGNAL(7)

NAME
       signal - list of available signals
...
 Standard Signals
       Linux supports the standard signals listed below. Several  signal  num-
       bers  are  architecture  dependent, as indicated in the "Value" column.
       (Where three values are given, the first one is usually valid for alpha
       and  sparc,  the  middle one for i386, ppc and sh, and the last one for
       mips.  A - denotes that a signal is absent on the corresponding  archi-
       tecture.)

       First the signals described in the original POSIX.1-1990 standard.

       Signal     Value     Action   Comment
       ------------------------------------------------------------------------
       SIGHUP        1       Term    Hangup detected on controlling terminal
                                     or death of controlling process
       SIGINT        2       Term    Interrupt from keyboard
       SIGQUIT       3       Core    Quit from keyboard
       SIGILL        4       Core    Illegal Instruction
       SIGABRT       6       Core    Abort signal from abort(3)
       SIGFPE        8       Core    Floating point exception
       SIGKILL       9       Term    Kill signal
       S ...
       S       SIGCHLD   20,17,18    Ign     Child stopped or terminated
       S       SIGCONT   19,18,25    Cont    Continue if stopped
       S       SIGSTOP   17,19,23    Stop    Stop process
       S       SIGTSTP   18,20,24    Stop    Stop typed at tty
       S       SIGTTIN   21,21,26    Stop    tty input for background process
               SIGTTOU   22,22,27    Stop    tty output for background process

               The  signals SIGKILL and SIGSTOP cannot be caught, blocked, or ignored.
```

lsm.: System V, BSD, POSIX jelkezelés

```
SIGNAL(2)                  Linux Programmer's Manual                 SIGNAL(2)

NAME
       signal - ANSI C signal handling

SYNOPSIS
       #include <signal.h>

       typedef void (*sighandler_t)(int);

       sighandler_t signal(int signum, sighandler_t handler);

DESCRIPTION
       The  signal()  system call installs a new signal handler for the signal
       with number signum.  The signal handler is set to sighandler which  may
       be a user specified function, or either SIG_IGN or SIG_DFL.
```

```
#include <stdio.h>
#include <signal.h>
int
main(void)
{
   signal(SIGINT, SIG_IGN);
   sleep(5);
   signal(SIGINT, SIG_DFL);
   for(;;)
     sleep(5);
   return 0;
}
```

PP 41

MM 382

Ism.: System V, **BSD**, POSIX jelkezelés

```
SIGNAL(2)                  Linux Programmer's Manual                 SIGNAL(2)

NAME
       signal - ANSI C signal handling

SYNOPSIS
       #include <signal.h>

       typedef void (*sighandler_t)(int);

       sighandler_t signal(int signum, sighandler_t handler);
...

RETURN VALUE
       The signal() function returns the previous value of the signal handler,
       or SIG_ERR on error.
```

```
#include <stdio.h>
#include <signal.h>
void utolso_tennivalo(int sig)
{
  printf("Utolso tennivalo kesz, immar kilephetek\a\n");
  exit(0);
}
int
main(void)
{
  if(signal(SIGINT, utolso_tennivalo) == SIG_IGN)
    signal(SIGINT, SIG_IGN);
  for(;;)
    putchar(getchar());
  return 0;
}
```

lsm.: System V, BSD, POSIX jelkezelés

```
SIGNAL(2)                  Linux Programmer's Manual                 SIGNAL(2)

NAME
       signal - ANSI C signal handling

SYNOPSIS
       #include <signal.h>

       typedef void (*sighandler_t)(int);

       sighandler_t signal(int signum, sighandler_t handler);

DESCRIPTION
       The  signal()  system call installs a new signal handler for the signal
       with number signum.  The signal handler is set to sighandler which  may
       be a user specified function, or either SIG_IGN or SIG_DFL.
```

```
#include <stdio.h>
#include <signal.h>
void ctrlc_kezelo(int sig)
{
  signal(SIGINT, ctrlc_kezelo);
  printf("Megprobaltal megallitani?\a\n");
}
int
main(void)
{
  if(signal(SIGINT, ctrlc_kezelo) == SIG_IGN)
    signal(SIGINT, SIG_IGN);
  for(;;)
    putchar(getchar());
  return 0;
}
```

PP 42

lsm.: System V, BSD, **POSIX** jelkezelés

```
SIGACTION(2)                Linux Programmer's Manual                SIGACTION(2)

NAME
       sigaction - examine and change a signal action

SYNOPSIS
       #include <signal.h>

...
      The sigaction structure is defined as something like

              struct sigaction {
                  void (*sa_handler)(int);
                  void (*sa_sigaction)(int, siginfo_t *, void *);
                  sigset_t sa_mask;
                  int sa_flags;
                  void (*sa_restorer)(void);
              }
```

```
#include <stdio.h>
#include <signal.h>
void ctrlc_kezelo(int sig)
{
  printf("Megprobaltal megallitani?\a\n");
}
int
main(void)
{
  struct sigaction sa;
  sa.sa_handler = ctrlc_kezelo;
  sigemptyset(&sa.sa_mask);
  sa.sa_flags = SA_RESTART;
  sigaction(SIGINT, &sa, NULL);
  for(;;)
    putchar(getchar());
  return 0;
}
```

Hol találkozol vele pl.?

crashme.c „újabb” verziók

agent2d-3.0.0/src/main_player.cpp

lsm.: Nem lokális ugrások

```
SETJMP(3)                    Library functions                    SETJMP(3)

NAME

SYNOPS
```

```
LONGJMP(3)                   Library functions                   LONGJMP(3)

NAME
       longjmp, siglongjmp - non-local jump to a saved stack context

SYNOPSIS
       #include <setjmp.h>
```

```
DE                                                    val);

                                                      for dealing with errors and inter-
..
```

```
#include <stdio.h>
#include <signal.h>
#include <setjmp.h>
sigjmp_buf jmpbuf;
void
kezdjuk_ujra (int sig)
{
  signal (SIGINT, kezdjuk_ujra);
  printf ("Megzavartal, ujra kezdjuk\a\n");
  siglongjmp (jmpbuf, 0);
}

int
main (void)
{
  if (signal (SIGINT, kezdjuk_ujra) == SIG_IGN)
    signal (SIGINT, SIG_IGN);
  sigsetjmp (jmpbuf, 1);
  printf("Kezdjuk!");
  for (;;)
    putchar (getchar ());
  return 0;
}
```

PP 42

```
$ gcc ugras.c -o ugras
$ ./ugras
Kezdjuk!alma
alma  Ctrl+C
Megzavartal, ujra kezdjuk
Kezdjuk!korte
korte Ctrl+C
Megzavartal, ujra kezdjuk
Kezdjuk!
[1]+  Stopped                 ./ugras
$ kill %1
$
[1]+  Terminated              ./ugras
```

Hogy öröklődésre, kivételkezelésre tudjunk példákat nézni jöjjön egy kis CORBA (részletesen majd a 10. előadásban foglalkozunk vele).

Mi a CORBA?

Elosztott OO infrastruktúra

1) Common Object Request Broker Architecture
2) Object Management Group (OMG)
3) Object Management Architecture (OMA)

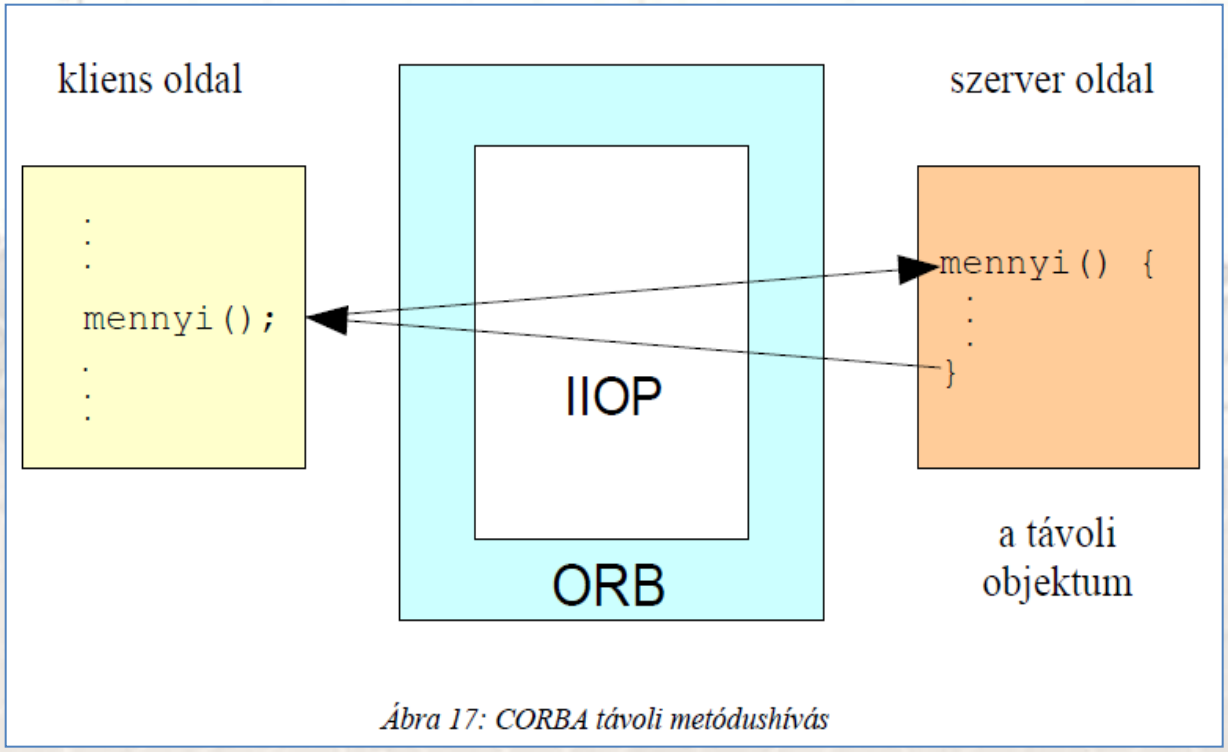
OMA architektúra:

1) OMG **IDL**: az objektummodell leírására (DEKLARATÍV NYELV!)
2) ORB
3) COS, objektumszolgáltatások, például: névszolgáltatás

Ism.: a CORBA világa

Common Object Request Broker Architecture

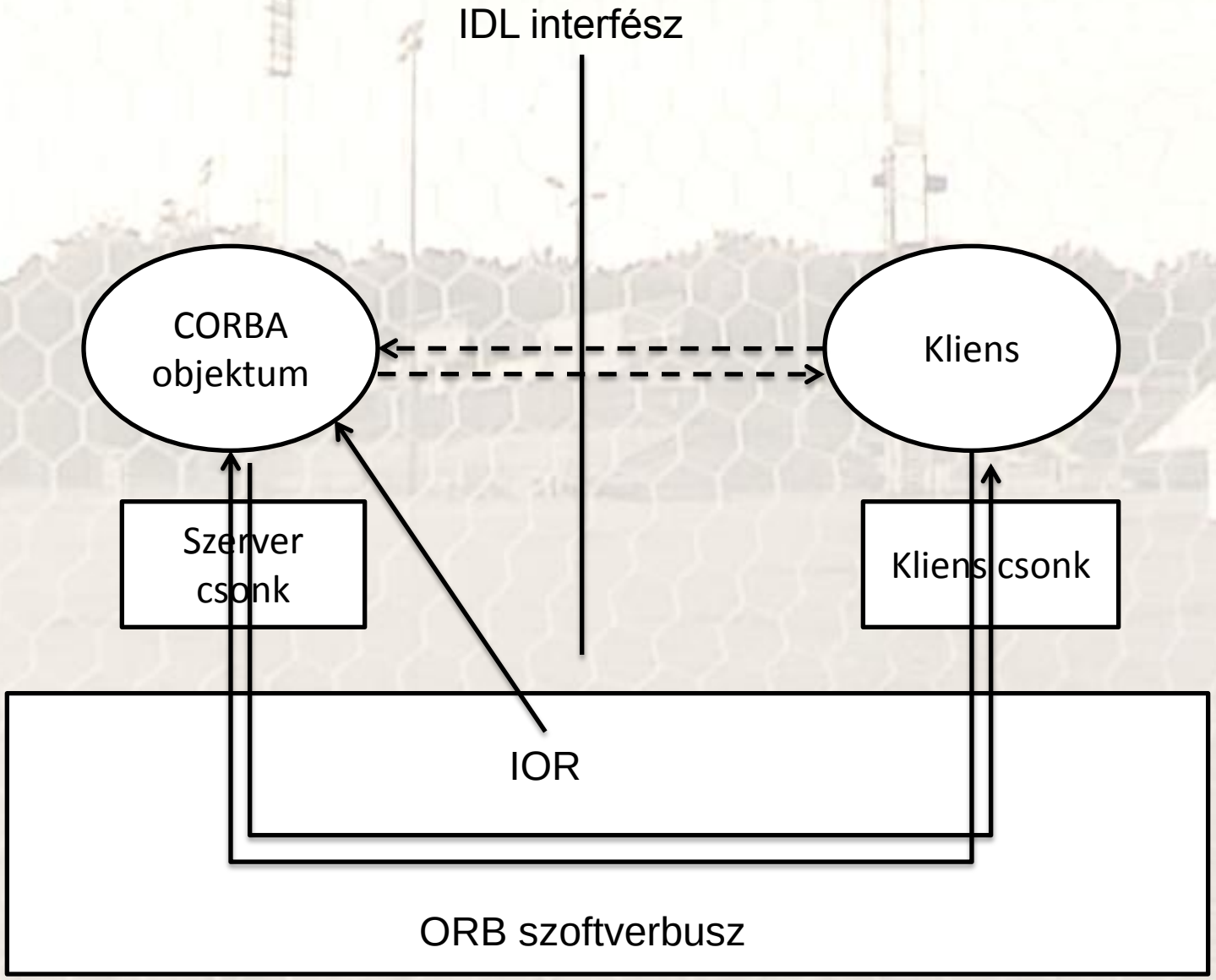
1) Elosztott
2) Heterogén
3) OO



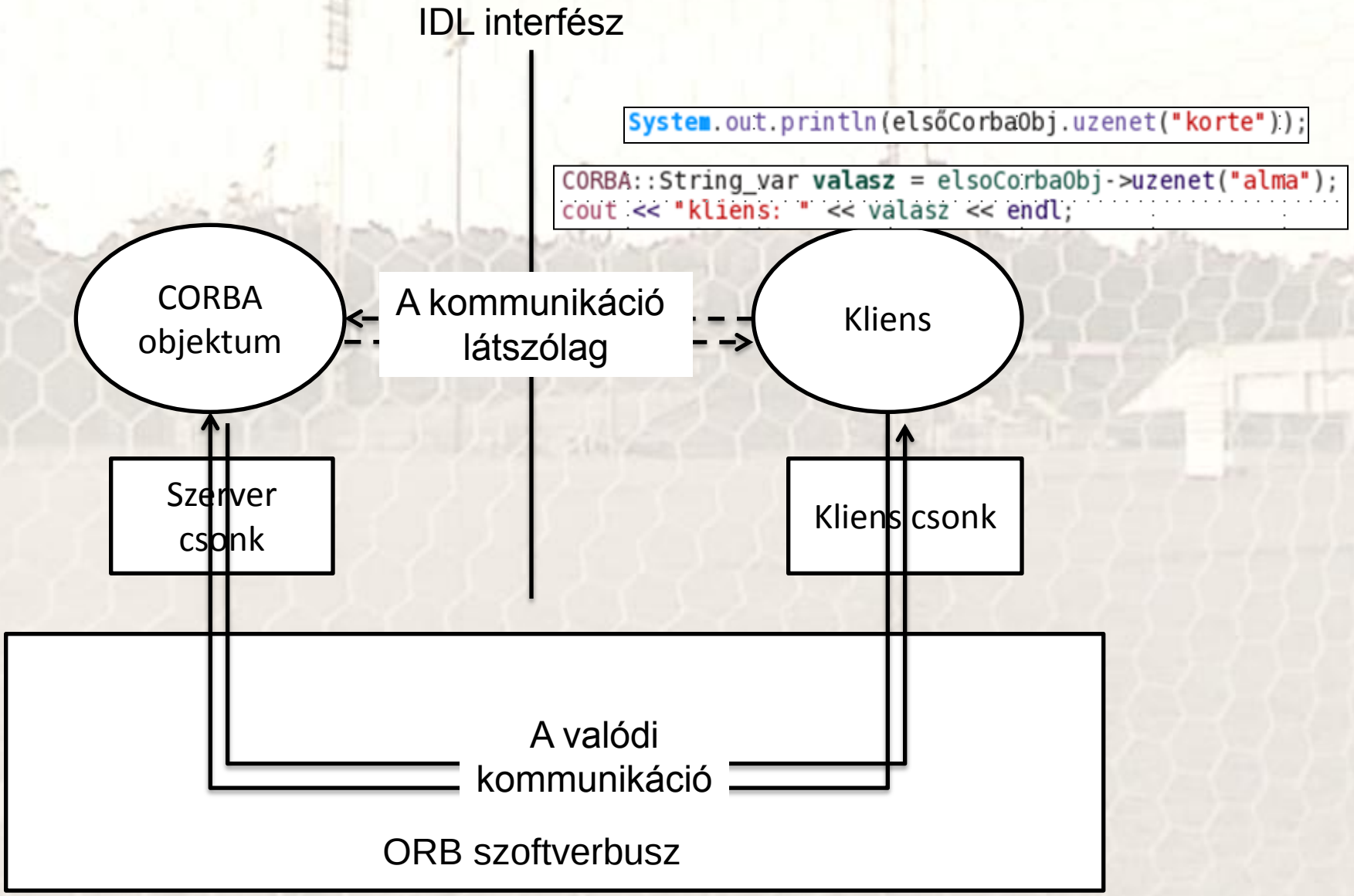
Ábra 17: CORBA távoli metódushívás

Object Management Group (OMG)
http://www.omg.org
CORBA 3.1 (2008)
http://www.omg.org/spec/CORBA/3.1
C Language Mapping Specification
http://www.omg.org/spec/C/1.0/PDF/
C++ Language Mapping, Version 1.2
http://www.omg.org/spec/CPP/1.2/PDF/

Az OMA architektúra



Az OMA architektúra



Az első CORBA objektum

```
interface ElsoCorbaObj {

  string uzenet(in string uzi);

};
```

```
class ElsoCorbaObj_i : public POA_ElsoCorbaObj
{

public:
    ElsoCorbaObj_i () {}
    ~ElsoCorbaObj_i () {}
    virtual char *uzenet (const char *u);
};

char *
ElsoCorbaObj_i::uzenet (const char *u)
{
    cout << "szerver: " << u << endl;
    return CORBA::string_dup (u);
}
```

Királyi út: csak a szolgáltatás (uzenet) implementálására kell koncentrálnia a fejlesztőnek.

OmniORB, http://omniorb.sourceforge.net/
omniidl -bcxx elso.idl

Java IDL
idlj -fall elso.idl

Az első CORBA objektum, szerver oldali részek

```
int
main (int argc, char *argv[])
{
    try
    {
```

1. Kapcsolat az ORB szoftverrel (tipikus paraméter: hoszt, port).
2. POA gyökerével (CORBA-ban is „minden objektum”, a típuskényszerítés eszköze a narrow, mert ugye a leszármazottak metódusait akarjuk majd hívni kell a cast)
3. Kiszolgáló objektumunk elkészítése
4. Névszolgáltató (helyi CORBA-s telefonkönyv „házi használatra”) elérése
5. Az „Elso” névvel összekötjük a kiszolgáló objektumunk referenciáját, így e név alapján szerezhetik majd meg a kliensek a referenciáját, hogy tudják „távoli” metódusát hívni

```
rbaObjektum);
    } catch (CORBA::Exception & e) {
    }
}
```

Nem értelmes, átgondolt kezelés, csak jelezzük, hogy gond volt és vége.

Az első CORBA objektum, szerver oldali részek

```
int
main (int argc, char *argv[])
{
    try
    {
        // Az ORB (a metódushívások közvetítőjének) inicializálása
        CORBA::ORB_var orb = CORBA::ORB_init (argc, argv);
        // A gyökér POA CORBA objektum referenciájának megszerzése
        CORBA::Object_var corbaObjektum = orb->resolve_initial_references ("RootPOA");
        // CORBA-s típuskényszerítés
        PortableServer::POA_var poa = PortableServer::POA::_narrow (corbaObjektum);
        // A kiszolgáló objektum létrehozása
        ElsoCorbaObj_i *elsoCorbaObj = new ElsoCorbaObj_i ();
        PortableServer::ObjectId_var elsoCorbaObjid = poa->activate_object (elsoCorbaObj);
        // CORBA objektum referencia elkészítése
        CORBA::Object_var elsoCorbaObjKisz = elsoCorbaObj->_this ();
        CORBA::String_var ior = orb->object_to_string (elsoCorbaObjKisz);
        cout << ior << endl;
        // A névszolgáltató gyökerének, mint CORBA objektum
        // referenciájának megszerzése
        corbaObjektum = orb->resolve_initial_references("NameService");
        // CORBA-s típuskényszerítés
        CosNaming::NamingContextExt_var rootContext = CosNaming::NamingContextExt::_narrow(corbaObjektum);
        // Kiszolgáló CORBA objektumom bejegyzése a névszolgáltatóba
        CosNaming::Name* name =  rootContext->to_name("Elso");
        rootContext->rebind(*name, elsoCorbaObjKisz);
        elsoCorbaObj->_remove_ref ();
        PortableServer::POAManager_var pman = poa->the_POAManager ();
        pman->activate ();
        orb->run ();
    } catch (CORBA::Exception & e) {
        cout << "CORBA kivétel: (CORBA::Exception) " << e._name () << endl;
    }
}
```

Az első CORBA objektum, kliens oldali részek

```
#include <iostream>
#include "elso.hh"

using namespace std;

int
main (int argc, char *argv[])
{
    try
    {
        // Az ORB (a metódushívások közvetítőjének) inicializálása
        CORBA::ORB_var orb = CORBA::ORB_init (argc, argv);
        orb->destroy();
    } catch (CORBA::Exception & e) {
    }
}
```

1. Kapcsolat az ORB szoftverrel (tipikus paraméter: hoszt, port).
2. Névszolgáltató (helyi CORBA-s telefonkönyv „házi használatra”) elérése
3. Az „Elso” névvel a kiszolgáló objektum referenciájának elkérése a névszolgáltató CORBA objektumtól
4. A távoli metódus (uzenet) meghívása

Nem értelmes, átgondolt kezelés, csak jelezzük, hogy gond volt és vé

Az első CORBA objektum, kliens oldali részek

```
#include <iostream>
#include "elso.hh"

using namespace std;

int
main (int argc, char *argv[])
{
    try
    {
        // Az ORB (a metódushívások közvetítőjének) inicializálása
        CORBA::ORB_var orb = CORBA::ORB_init (argc, argv);
        // A névszolgáltató gyökerének, mint CORBA objektum
        // referenciájának megszerzése
        CORBA::Object_var corbaObjektum;
        corbaObjektum = orb->resolve_initial_references("NameService");
        // CORBA-s típuskényszerítés
        CosNaming::NamingContextExt_var nevszolgaltatoGyokere =
            CosNaming::NamingContextExt::_narrow(corbaObjektum);
        // Az "ElsoCorbaObj" (visszhang) szolgáltatást nyújtó CORBA objektum
        // referenciájának megszerzése
        CORBA::Object_ptr obj = nevszolgaltatoGyokere->resolve_str("Elso");
        // CORBA-s típuskényszerítés
        ElsoCorbaObj_var elsoCorbaObj = ElsoCorbaObj::_narrow(obj);
        // a szolgáltatás igánybe vétele:
        CORBA::String_var valasz = elsoCorbaObj->uzenet("alma");
        cout << "kliens: " << valasz << endl;
        // ORB leállítása
        orb->destroy();
    } catch (CORBA::Exception & e) {
        cout << "CORBA kivétel: (CORBA::Exception) " << e._name () << endl;
    }
}
```

Az első CORBA objektum, kliens oldali részek (Java-ban)

```
public class Kliens {

    public static void main(String[] args) {

        try {
            // Az ORB (a metódushívások közvetítőjének) inicializálása
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

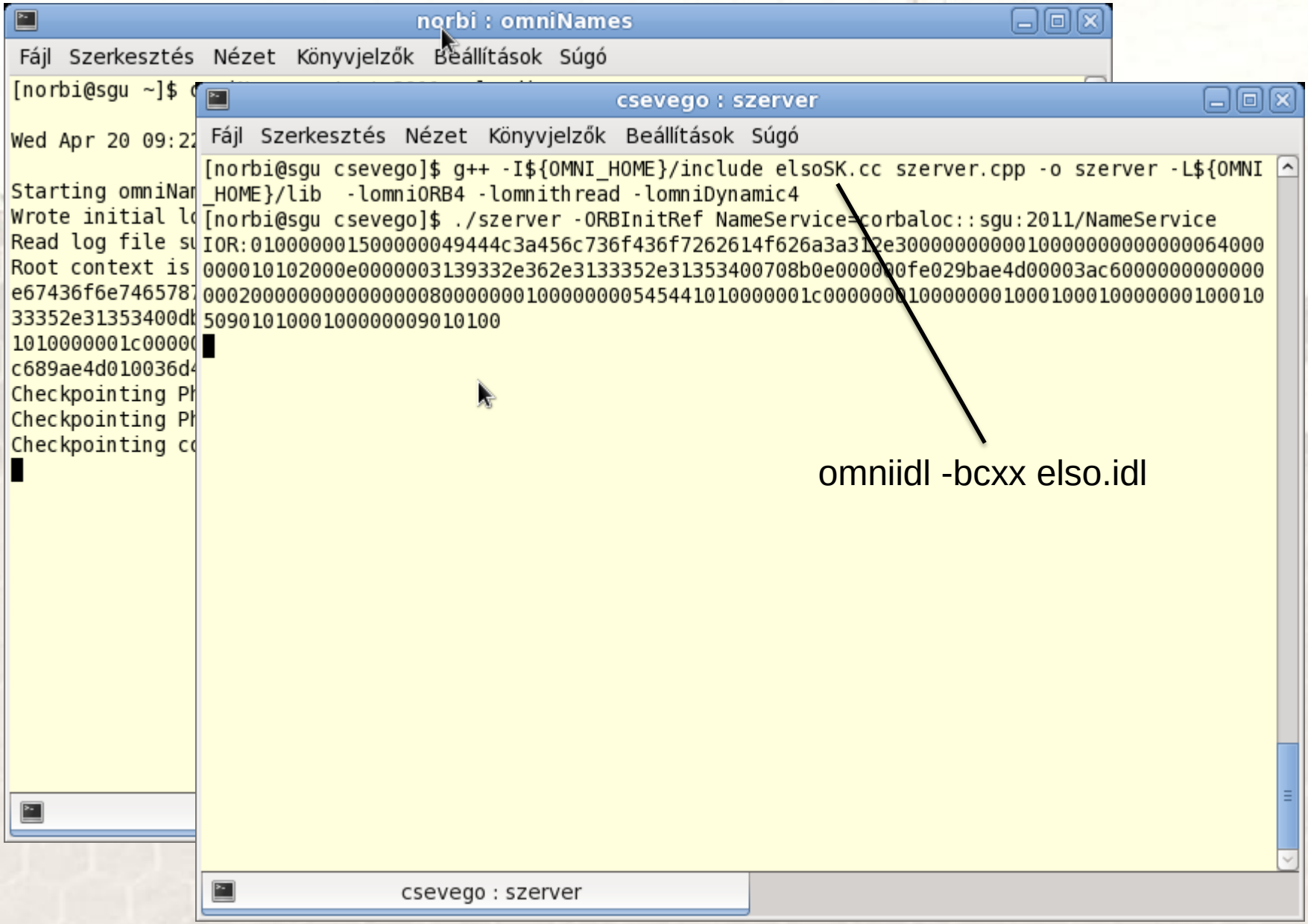
1. Kapcsolat az ORB szoftverrel (tipikus paraméter: hoszt, port).
2. Névszolgáltató (helyi CORBA-s telefonkönyv „házi használatra”) elérése
3. Az „Elso” névvel a kiszolgáló objektum referenciájának elkérése a névszolgáltató CORBA objektumtól
4. A távoli metódus (uzenet) meghívása

Nem értelmes, átgondolt kezelés, csak jelezzük, hogy gond volt és v

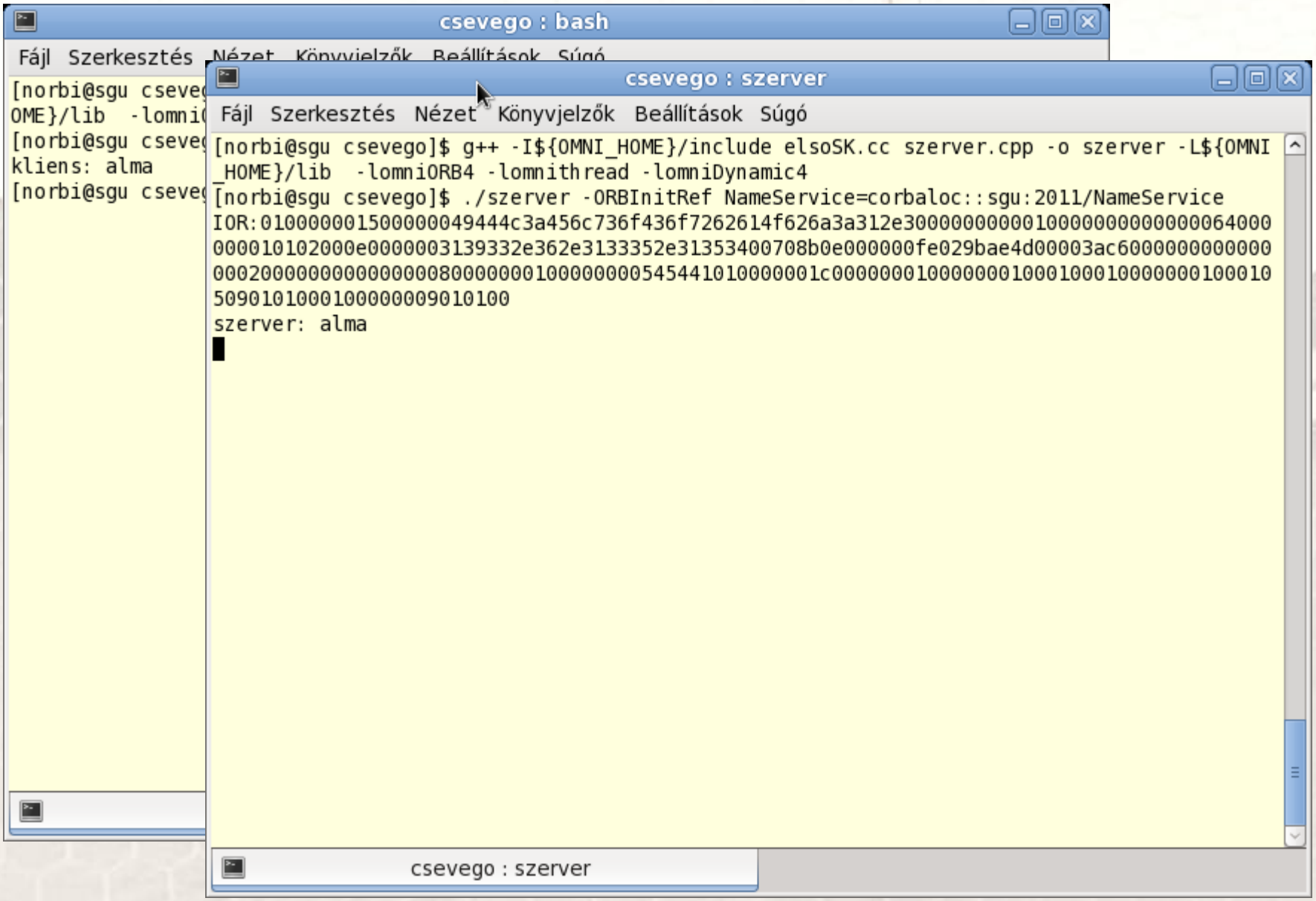
Az első CORBA objektum, kliens oldali részek (Java-ban)

```
public class Kliens {

    public static void main(String[] args) {

        try {
            // Az ORB (a metódushívások közvetítőjének) inicializálása
            org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args, null);
            // A névszolgáltató gyökerének, mint CORBA objektum
            // referenciájának megszerzése
            org.omg.CORBA.Object corbaObjektum =
                orb.resolve_initial_references("NameService");
            org.omg.CosNaming.NamingContextExt névszolgáltatóGyökere
            = org.omg.CosNaming.
              NamingContextExtHelper.narrow(corbaObjektum);
            // Az "ElsoCorbaObj" (visszhang) szolgáltatást nyújtó CORBA objektum
            // referenciájának megszerzése
            ElsoCorbaObj elsőCorbaObj =
                ElsoCorbaObjHelper.narrow(
                    névszolgáltatóGyökere.resolve_str("Elso"));
            // a szolgáltatás igánybe vétele:
            System.out.println(elsőCorbaObj.uzenet("korte"));

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

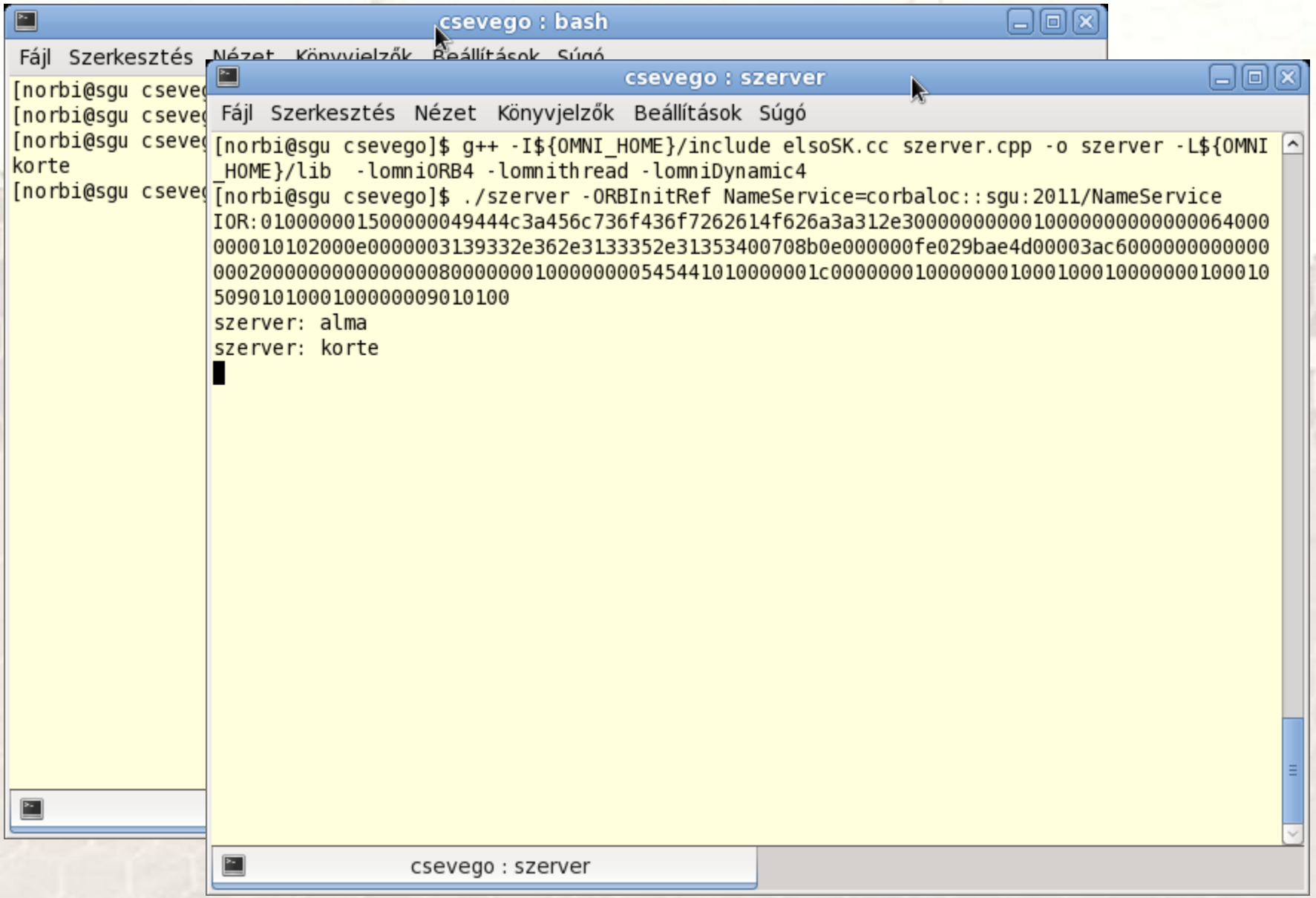
Az első CORBA objektum tesztelése



Az első CORBA objektum, C++ kliens



Az első CORBA objektum, Java kliens



Vissza a kivételkezeléshez

Lásd majd későbbi laboron (és 10. ea.): nem a NamingContextExt-et használva: bejegyezni egy kontextust, ha már van: kivétel...

CosNaming::NamingContext::AlreadyBound
Kezelése: használjuk, ami már hozzá van kötve

A kiszolgáló objektum már hozzá van kötve kivétel…
CosNaming::NamingContext::AlreadyBound
Kezelése: rebind

Illetve érdemes lesz játszani majd: elírni a neveket és kiírni milyen CORBA kivétel jön stb.

Vissza a kivételkezeléshez

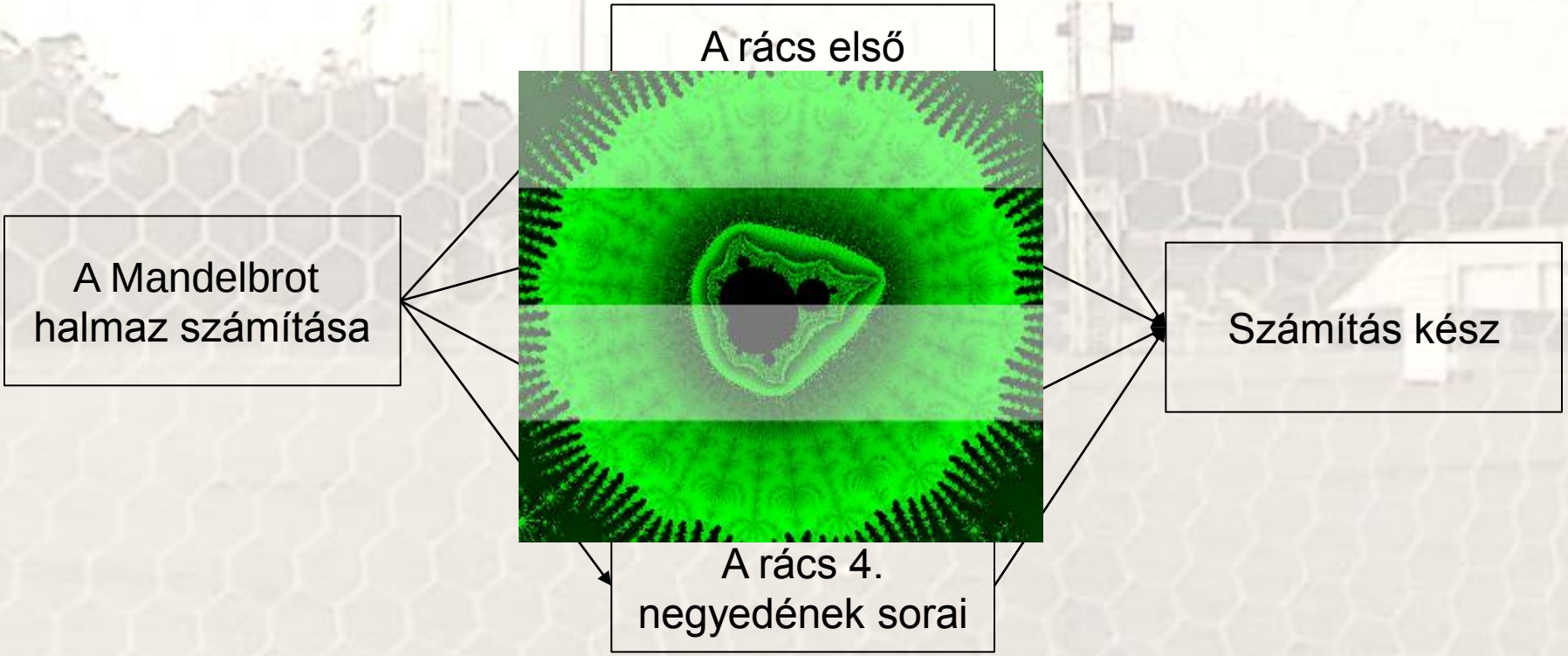
```
try {
        Osztaly1 peldany1;
        Osztaly2 peldany2 = new Osztaly2();
        …
        throw ”hiba”;
        …
        delete peldany2;
} catch (char * hiba) {
        …
}
```

Mi a helyzet a peldany?-ekkel?

Vissza a kivételkezeléshez

```
try {
        …
        throw KivetelOsztaly();
        …
} catch (KivetelOsztaly vagy annak őse) {
        …
        ha nem tudja kezelni, tovább dobhatja: throw
        …
} catch (…) {

}
```

Labor, párhuzamos programozás



Párhuzamos programozás, P-szálak

A rács első negyedének sorai

A rács 2.

A Mande halmaz szá

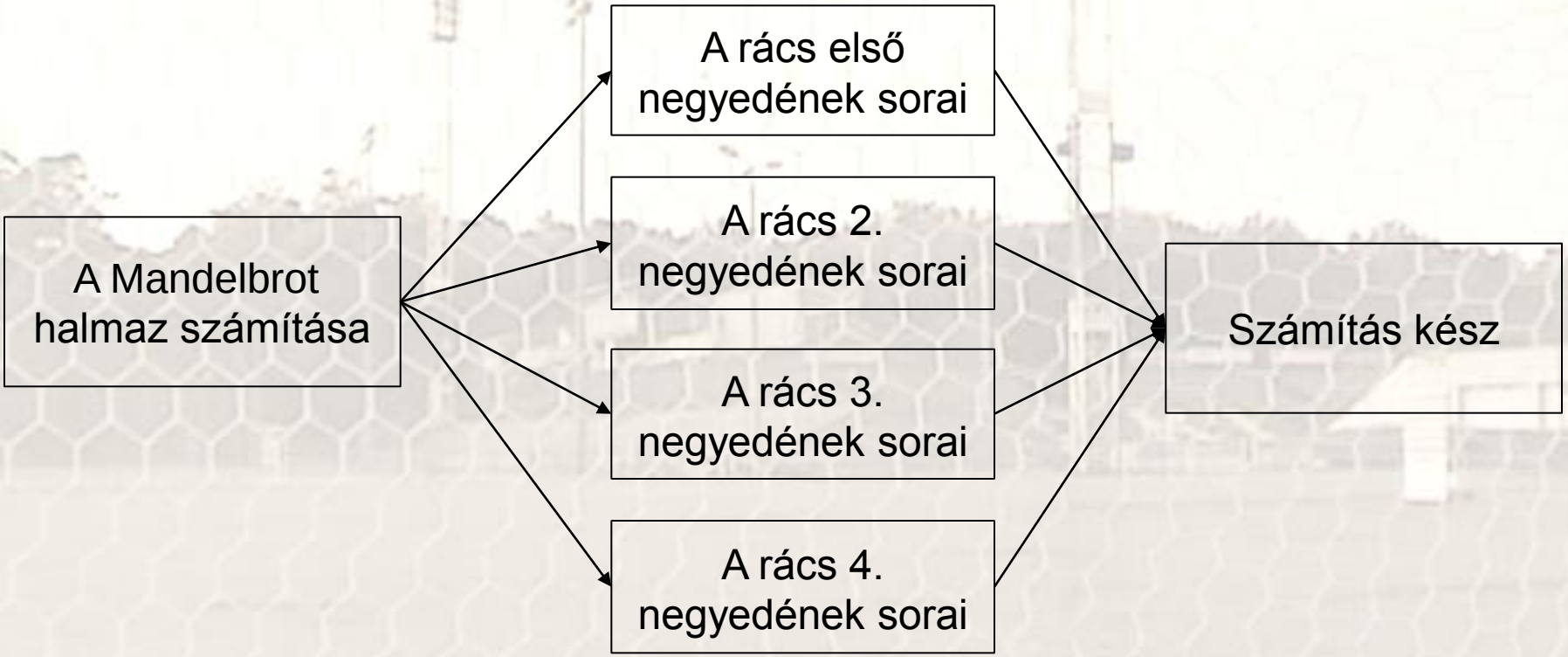
tás kész

```
void *mandel_resz_szamitas(void *id)
{
    int mettol = *(int *)id * ( magassag / SZALAK_SZAMA);
    int meddig = mettol + (magassag/SZALAK_SZAMA);

    // a számítás
    double dx = (b-a)/szelesseg;
    double dy = (d-c)/magassag;
    double reC, imC, reZ, imZ, ujreZ, ujimZ;
    // Hány iterációt csináltunk?
    int iteracio = 0;
    std::cout << *(int *)id << ". szal szamitas indul";
    // Végigzongorázzuk a szélesség x magasság rácsot:
    for (int j=mettol; j<meddig; ++j) {
        //sor = j;
        for (int k=0; k<szelesseg; ++k) {
            // c = (reC, imC) a rács csomópontjainak
            // megfelelő komplex szám
```

http://progpater.blog.hu/2011/03/27/a_parhuzamossag_gyonyorkodtet

Párhuzamos programozás, OpenMP

Open Multi-Processing

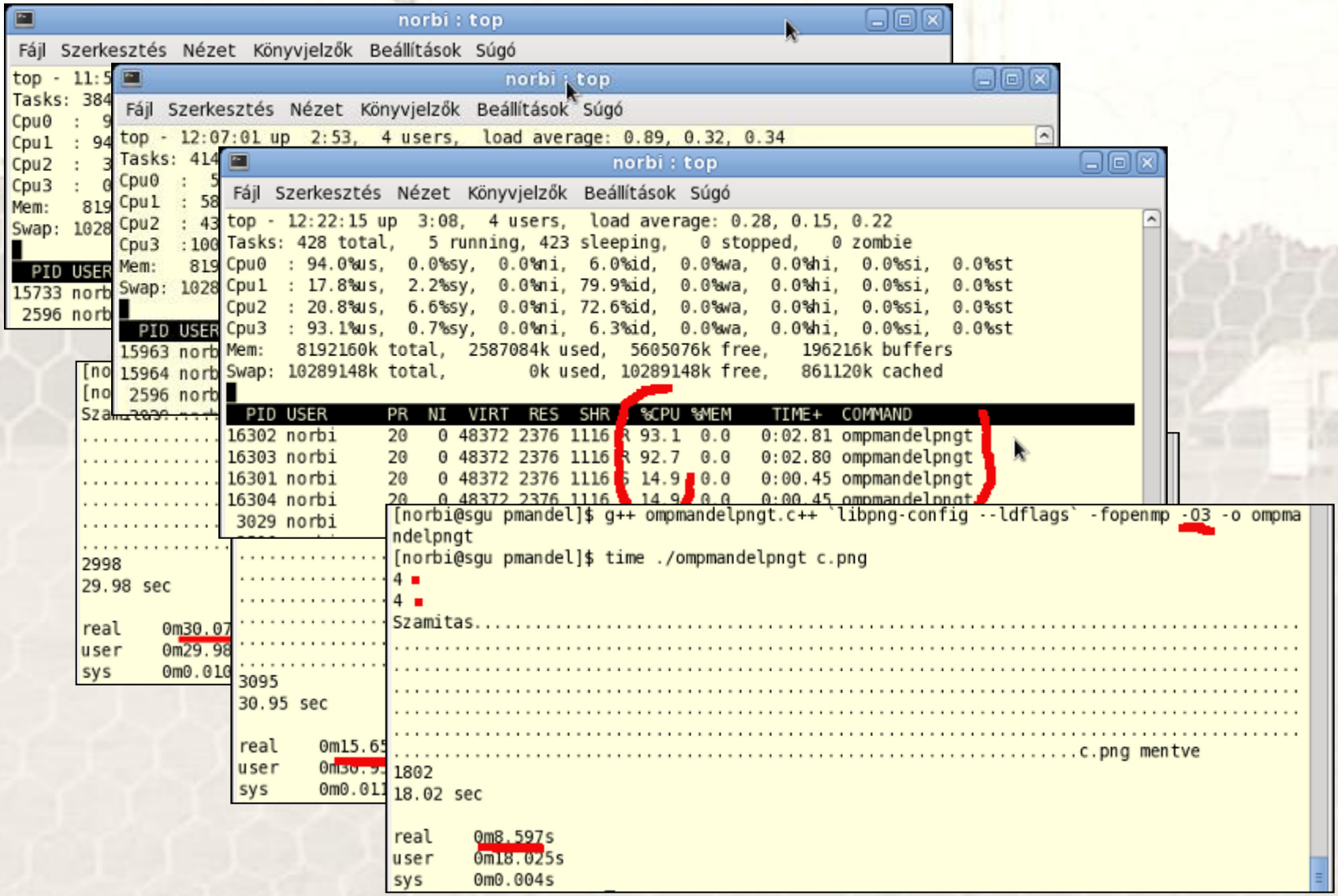



```
// Végigzongorázzuk a szélesség x magasság rácsot:
#pragma omp parallel for
for (int j=0; j<magassag; ++j) {
    //sor = j;
    for (int k=0; k<szelesseg; ++k) {
        // c = (reC, imC) a rács csomópontjainak
        // megfelelő komplex szám
        reC = a+k*dx;
```

Ciklus párhuzamosítása

http://progpater.blog.hu/2011/03/27/a_parhuzamossag_gyonyorkodtet

Szekvenciális vs. párhuzamos



http://progpater.blog.hu/2011/03/27/a_parhuzamossag_gyonyorkodtet

Szekvenciális vs. párhuzamos

```
[norbi@sgu exor]$ gcc t.c -O3 -o t -std=c99
[norbi@sgu exor]$ time ./t <titkos.szoveg |grep 59934434
59934434Már komoly mennyiségű munkát elvégeztünk ebben a
félévben (eddig is, de még sok élmény vár :), ezért aki nem
készült heti gyakorisággal, az úgy érezheti: elmélyült körülötte
a víz. Itt a tavaszi szünet holnaptól, az érintve érzett
hallgatóknak azt ajánlom, hogy pótolják be az elmaradt munkát,
élményeket, tapasztal
a blognak az elején
ahol (lehetőleg) gyor
kérdésekre! Ha a
elmélkedésbe kezdenél

real    3m34.691s
user    3m33.855s
sys     0m0.069s
```

```
[norbi@sgu exor]$ gcc t.c -fopenmp -O3 -o t -std=c99
[norbi@sgu exor]$ time ./t <titkos.szoveg |grep 59934434
59934434Már komoly mennyiségű munkát elvégeztünk ebben a
félévben (eddig is, de még sok élmény vár :), ezért aki nem
készült heti gyakorisággal, az úgy érezheti: elmélyült körülötte
a víz. Itt a tavaszi szünet holnaptól, az érintve érzett
hallgatóknak azt ajánlom, hogy pótolják be az elmaradt munkát,
élményeket, tapasztalatokat, azaz programozzanak. Kezdjék ennek
a blognak az elején és fogják fel olyan on-line tutoriálnak,
ahol (lehetőleg) gyorsan választ is adunk a kommentben felmerült
kérdésekre! Ha a programozás helyett a kurzust átszervező
elmélkedésbe kezdenél, akkor kezd ezzel :)

real    1m10.466s
user    3m50.372s
sys     0m0.057s
```

http://progpater.blog.hu/2011/03/28/a_hetedik_nyolcadik_labor

Párhuzamos programozás, OpenMP

Miért nem működik ugyanaz a ciklus párhuzamosító megoldás alapban, mint ami a Mandelbrot halmaz számítása esetében működött?

http://progpater.blog.hu/2011/03/28/a_hetedik_nyolcadik_labor

Labor, Qt kódolási konvenciók

1) Osztály nevek (class FrakSzal)
2) Metódus nevek (int szomszedokSzama(bool **racs,
   int sor, int oszlop, bool allapot);)
3) Azonosítók, konstansok, tagok (static const bool ELO = true;
   bool ***m_Racsok;)
4) Lekérdező/beállító módszerek
   racsok() - getRacsok() / setRacsok()

stb.

(OSS könyv)
Alan Ezust, Paul Ezust: ***An Introduction to Design Patterns in C++ with Qt 4***, Prentice Hall (Open Source Series) 2006

Pdf-ben:
http://ptgmedia.pearsoncmg.com/images/9780131879058/downloads/0131879057_Ezust_book.pdf

90. oldal.

Kódolási konvenciók általában

Robotfoci

Robotfoci

- RoboCup Soccer
  - **Soccer Simulation League**
    - **2D**
    - 3D
  - Soccer Humanoid League, az itt használt[ül is megkülönböztetnek három alkategóriát,[t lehet gyerek, ifjúsági vagy felnőtt. A két most e[ljuk méret szerint növekvő sorrendben a **Soc**[ol maximum 15 cm magas és 18 cm átmérő[csatáznak. Majd a **Soccer Middle Size Lea**[ternél kisebb átmérőjűek. Itt említjük a **Standar**[ahol egységesen ugyanazokat a robotokat has[megfelelően 2008-ig beszélhettünk a **Sor**[ú” ligáról. Azóta és napjainkban pedig az **Al**[s [KALYANAKRISHNAN]. Ezek a robotok két[centis humanoidok.

http://en.wikipedia.org/wiki/File:RUNSWift_Nao's_2010.jpg

= http://www.inf.unideb.hu/~nbatfai/book.pdf

Robotfoci

- **RoboCup Rescue**: itt mentést végző robotokkal találkozhatunk, ezen belül is valódiakkal (Rescue Robot League) vagy szimuláltakkal (Rescue Simulation League).
- **RoboCup @Home**: ha valahová a klasszikus porszívó robotot akarjuk elhelyezni, akkor ide kell tennünk, de ne becsüljük le ezt a kategóriát, hiszen itt lesz majd az „I, robot” film [IROBOT] Sonny-ja is!
- **RoboCup Junior**: e kategórián belül is találkozunk foci és mentő robotokkal, illetve újdonságként megjelenik a táncoló (robodance) robot. A kategória legfőbb jellemzője a célkorosztály lehetőségeinek megfelelően a LEGO© robotok használata, hiszen a LEGO Mindstorms® Robotics Invention System (RIS 2.0) csomagot a gyártó 12 áves kortól ajánlotta. A várhatóan 10 éves termékciklusát nemrégiben megkezdő új NXT csomagnál ezt a határt már a 10 éves korra szállították le.

 = http://www.inf.unideb.hu/~nbatfai/book.pdf

Robotfoci

RoboCup 2011: Adult Size Final

http://www.youtube.com/watch?v=llfYoFG7WrY

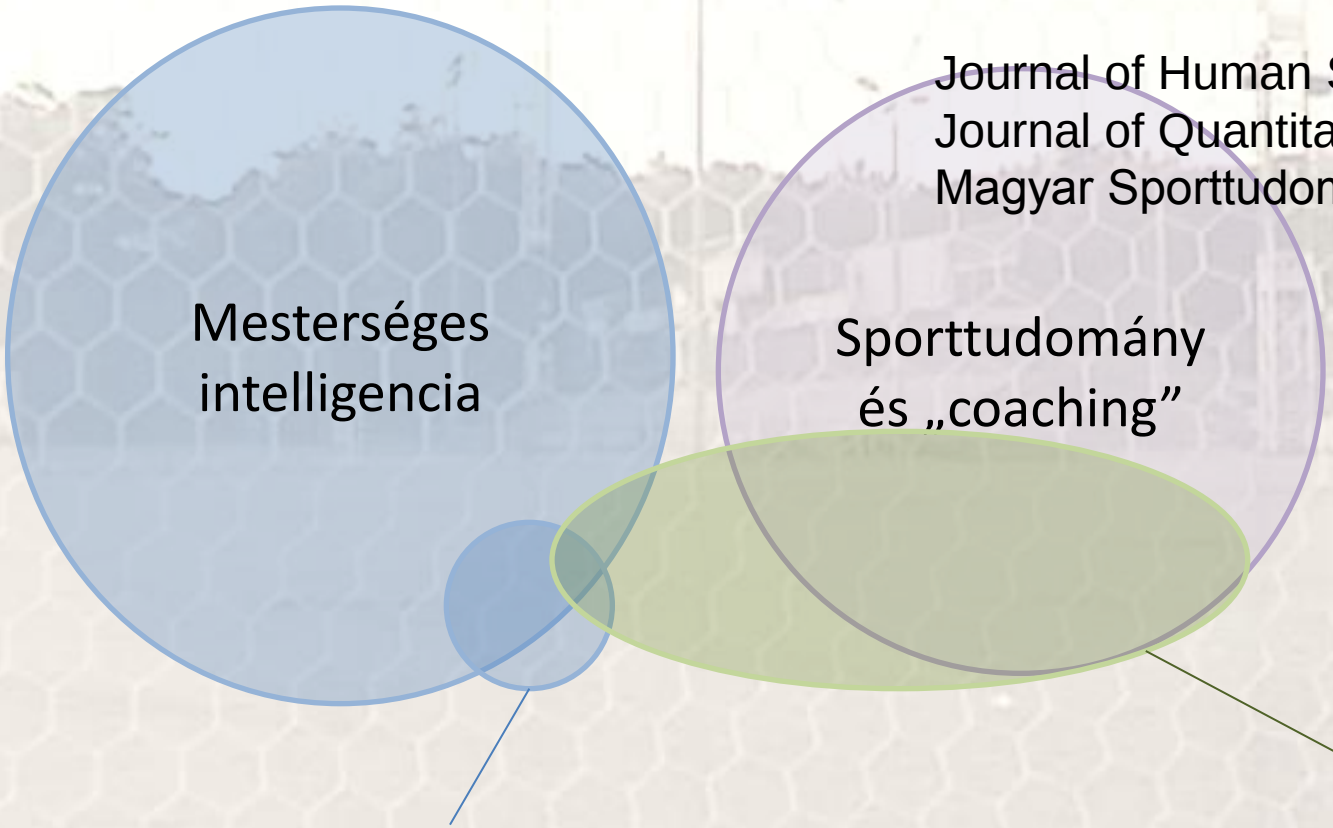
Ism: Robotfoci vs. FerSML

Miért alkalmatlan a RC a mi (FerSML sporttudományi) céljainkra?

Jóval magasabb absztrakciós szinten mozgunk: például a piramis alapú üzem nálunk egy belépési pont, amit a robot focinál kialakítani már komoly eredmény. (Mi nem akarunk a semmiből egy olyan játékos ágenst kialakítani, aki rendelkezik a pálya és a játék egy belső reprezentációjával, így képes intelligens viselkedésre, mert triviálisan feltesszük, hogy ez adott.)

(Aki elkészíti saját robotfocis csapatát, tapasztalni fogja, hogy ugyanaz a fejlesztői élmény, mintha csak egy LEGO robotot programozott volna.)

Ism: FerSML irodalomkutatás és célkitűzés



Sport Science Journals:

Journal of Human Sport and Exercise
Journal of Quantitative Analysis in Sports
Magyar Sporttudományi Szemle

Ism: RoboCup Soccer 2D Simulation League

Alapcikk: Hi and Eiichi
Osawa. 199 dings of the
first internat ACM, New
York, NY, US

Team

11+1 cl

n Other

clients

Monitor

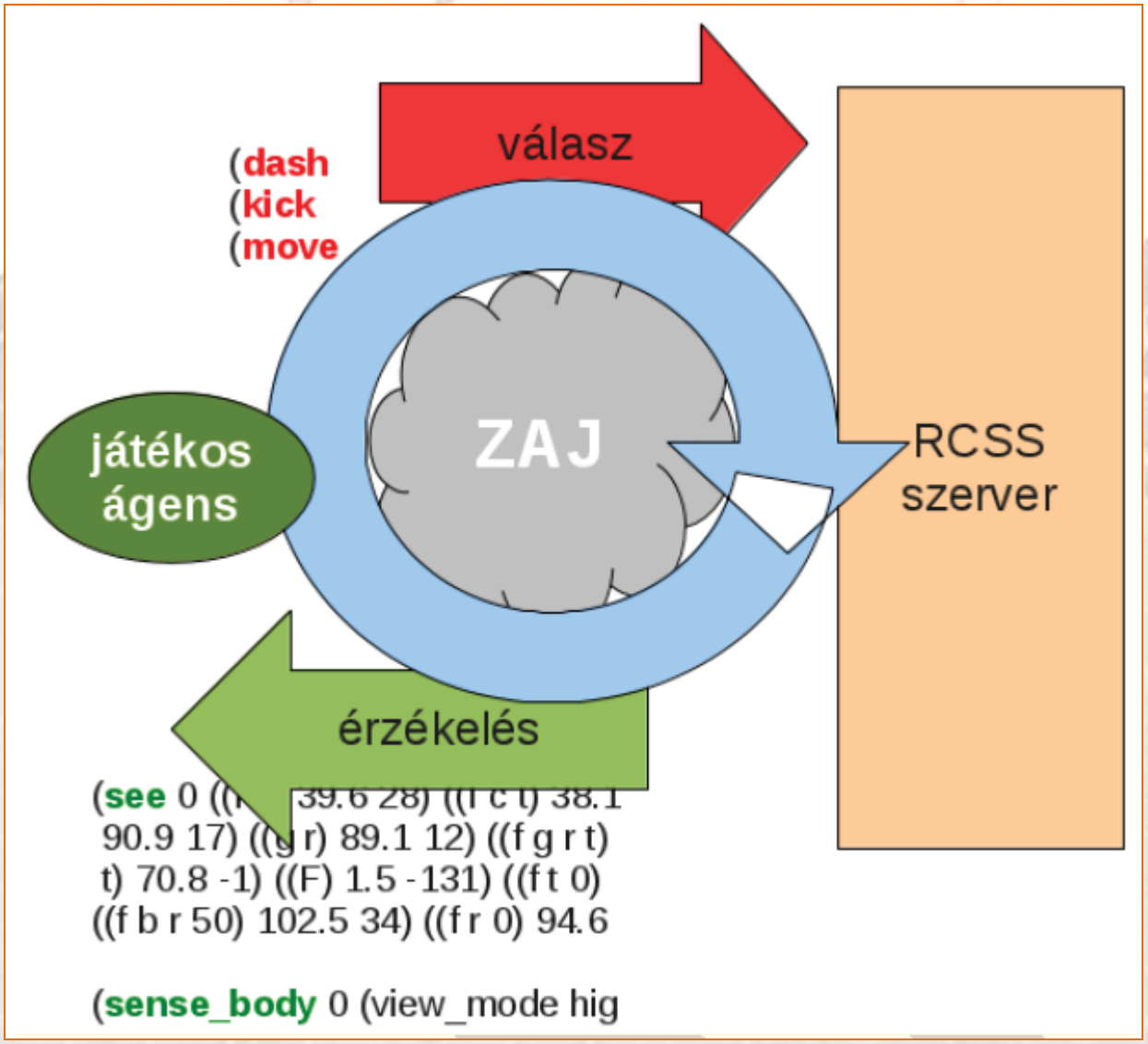
Ism: The RoboCup Soccer Simulator (RCSS) rcssserver

https://sourceforge.net/projects/sserver/

Egy szimulációs lépés:

6000 lépés 10 percben.

Az RCSS szimulációs ciklus



Robotfoci, szoftverek

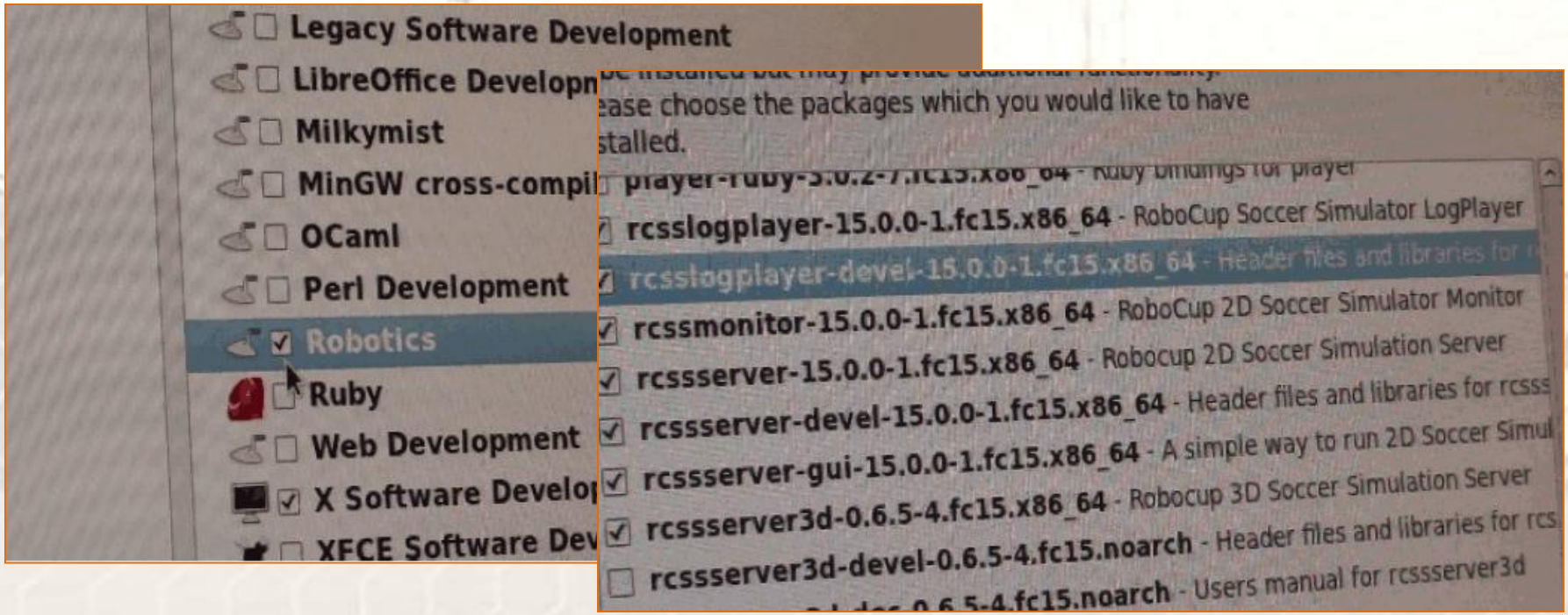
1.2. A kapcsolódó szoftverek és dokumentáció bemutatása

A jelen jegyzetben három szoftver (esetenként szoftver-csokor) megismerésére koncentrálunk, ezek a

- szimulációt megvalósító **RoboCup Soccer Simulator**, http://sourceforge.net/projects/sserver/ .
  - rcssserver, https://sourceforge.net/projects/sserver/files/rcssserver/ .
  - rcssmonitor, https://sourceforge.net/projects/sserver/files/rcssmonitor/ .
  - rcssmanual, https://sourceforge.net/projects/sserver/files/rcssmanual/ .
  - rcsslogplayer, https://sourceforge.net/projects/sserver/files/rcsslogplayer/ .
- a C++ alapú **Agent2D** kliens, http://en.sourceforge.jp/projects/rctools/ .
  - agent2d, http://en.sourceforge.jp/projects/rctools/releases/?package_id=4887 .
  - SoccerWindow, http://en.sourceforge.jp/projects/rctools/releases/?package_id=1917 .
  - soccerwindow2, http://en.sourceforge.jp/projects/rctools/releases/?package_id=4886 .
  - FormationEditor, http://en.sourceforge.jp/projects/rctools/releases/?package_id=11389.
- és a Java alapú **Atan** kliens interfész, http://sourceforge.net/projects/atan1/ .
  - atan.jar, 0.4.3, https://sourceforge.net/projects/atan1/files/Atan/ .
  - atan.jar, 1.0, svn co https://atan1.svn.sourceforge.net/svnroot/atan1 atan1 (illetve Maven projektté alakítva egyetlen paranccsal le tudjátok gyártani az atan-1.0.0.jar állományt).

 = http://www.inf.unideb.hu/~nbatfai/book.pdf

Rcssserver telepítés



RCSS, rcssserver

1.2.1. RoboCup Soccer Simulator

1.2.1.1. rcssserver

Az rcssserver a foci világának „mátrixa”, karakteres felületű szerver folyamat. Fejlesztése 1997 óta folyamatos, licence GNU LGPL.

RCSS, rcssmonitor

1.2.1.2. rcssmonitor

Az rcssmonitor feladata a szerver által felépített, karbantartott szimulációs világ megjelenítése. Fejlesztése a kezdetektől a rcssserver-el összefonva történik, licence GNU GPL v3.

RCSS, rcsslogplayer

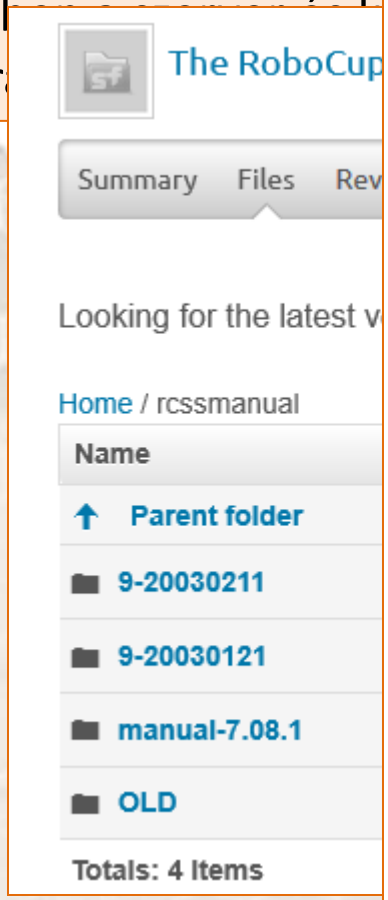
1.2.1.3. rcsslogplayer

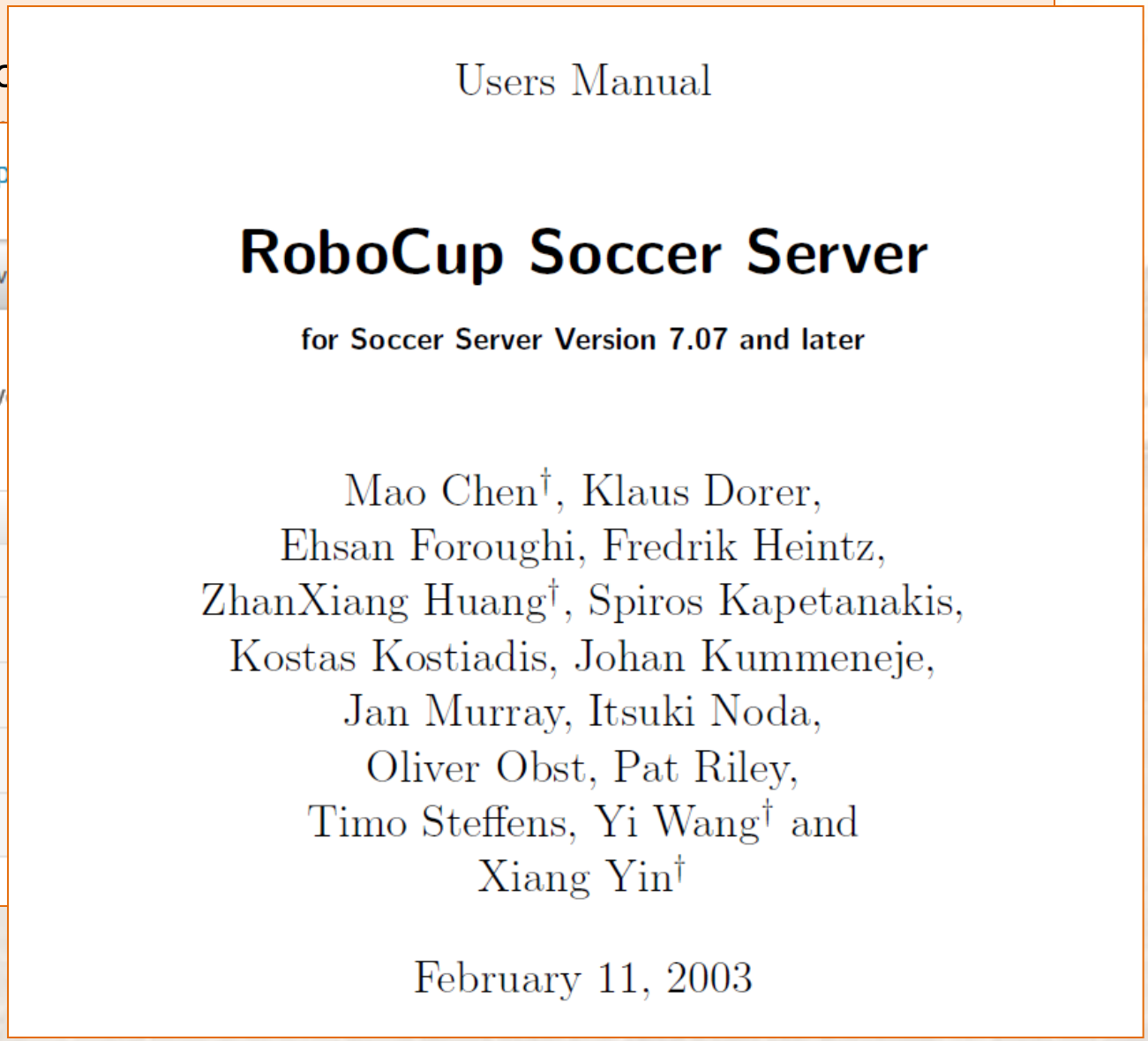
Az RCSS szerver alapértelmezésben menti abba a könyvtárba, ahonnan elindították a mérkőzés **rcg** állományát, az rcsslogplayer képes ezt rcssmonitor programként visszajátszani, licence GNU GPL v3.

RCSS, rcssmanual

1.2.1.4. rcssmanual

Az rcssmanual a Robo
egy
forr




Users Manual

RoboCup Soccer Server

for Soccer Server Version 7.07 and later

Mao Chen†, Klaus Dorer,
Ehsan Foroughi, Fredrik Heintz,
ZhanXiang Huang†, Spiros Kapetanakis,
Kostas Kostiadis, Johan Kummeneje,
Jan Murray, Itsuki Noda,
Oliver Obst, Pat Riley,
Timo Steffens, Yi Wang† and
Xiang Yin†

February 11, 2003

RoboCup tools

hogy el tudjuk helyezni, milyen a HELIOS

RoboCup Champions

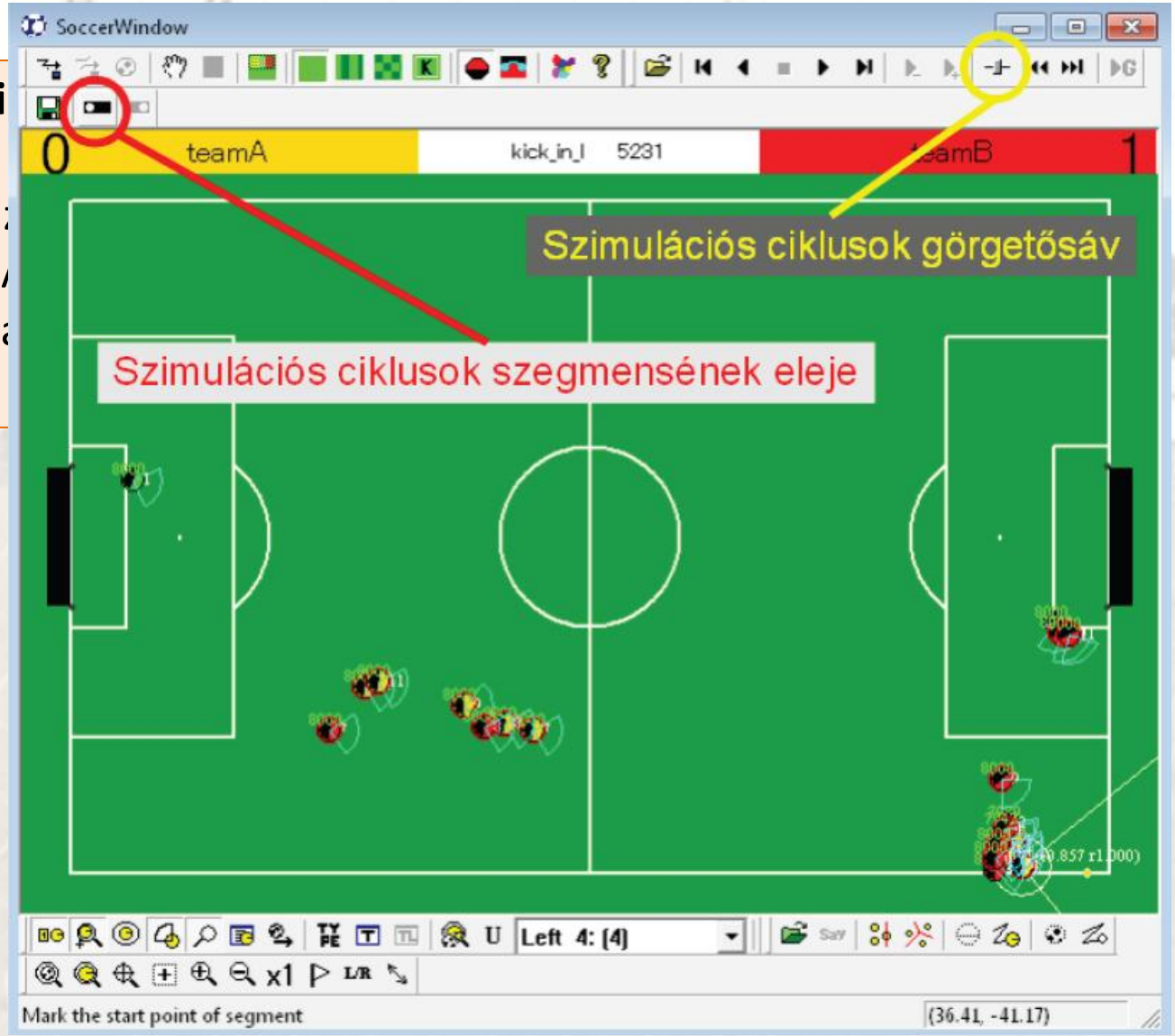
Year	1st Place Team	2nd Place Team	3rd Place Team	Media
1997	AT Humboldt, Germany	Andhill, Japan	ISIS, USA	Final
1998	CMUnited, USA	AT Humboldt, Germany	WindmillWanderer, The Netherlands	Final
1999	CMUnited, USA	magmaFreiburg, Germany	Essex Wizards, UK	Final
2000	FC Portugal, Portugal	Brainstormers, Germany	ATTCMUnited, USA	Final
2001	TsinghuAeolus, China	Brainstormers, Germany	FC Portugal, Portugal	Final1 Final2
2002	TsinghuAeolus, China	Everest, China	Brainstormers, Germany	Final
2003	UvA Trilearn, The Netherlands	TsinghuAeolus, China	Brainstormers, Germany	Final
2004	STEP, Russia	Brainstormers, Germany	Mersad, Iran	Final
2005	Brainstormers, Germany	WrightEagle, China	TokyoTechSFC, Japan	Final
2006	WrightEagle, China	Brainstormers, Germany	Ri-one, Japan	Final
2007	Brainstormers, Germany	WrightEagle, China	HELIOS, Japan	Final
2008	Brainstormers, Germany	WrightEagle, China	HELIOS, Japan	Final1 Final2
2009	WrightEagle, China	HELIOS, Japan	Oxsy, Romania	Final
2010	HELIOS, Japan	WrightEagle, China	Oxsy, Romania	Final
2011	WrightEagle, China	HELIOS, Japan	MarliK, Iran	Final

http://sourceforge.net/apps/mediawiki/sserver/index.php?title=Main_Page

RoboCup tools, SoccerWindow, soccerwindow2

1.2.2.3. SoccerWi

Mindkét program
programok is, aza
adott mérkőzést. A
funkciót (például a
is biztosít.



RoboCup tools, agent2d

1.2.2.2. agent2d

Egy példa RCSS kliens [HELIOS], licence ugyancsak GNU GPL v3. A fejlesztők külön kiemelik a kapcsolódó éves TDPkben, például a [HELIOS]-ban, hogy kezdő csapatoknak ideális lehet az elinduláshoz ezt választani a RoboCup-on való sikeres szerepléshez. A 2011-es torna résztvevői közül (természetesen a fejlesztő, most második helyet szerző HELIOS csapaton túl) a EdInferno.2D [EDINFERNO2D], ParaNoid [PARANOID], NADCO-2D [NADCO2D], AUA2D [AUA2D], Photon csapatok fogadták meg ezt a tanácsot.

```
[norbert@matrica RoboCup]$ mv ../Downloads/agent2d-3.1.0.tar.gz .
[norbert@matrica RoboCup]$ gunzip agent2d-3.1.0.tar.gz
[norbert@matrica RoboCup]$ tar xvf agent2d-3.1.0.tar
[norbert@matrica RoboCup]$ cd agent2d-3.1.0
[norbert@matrica agent2d-3.1.0]$ ./configure
[norbert@matrica agent2d-3.1.0]$ make
```

RoboCup2011

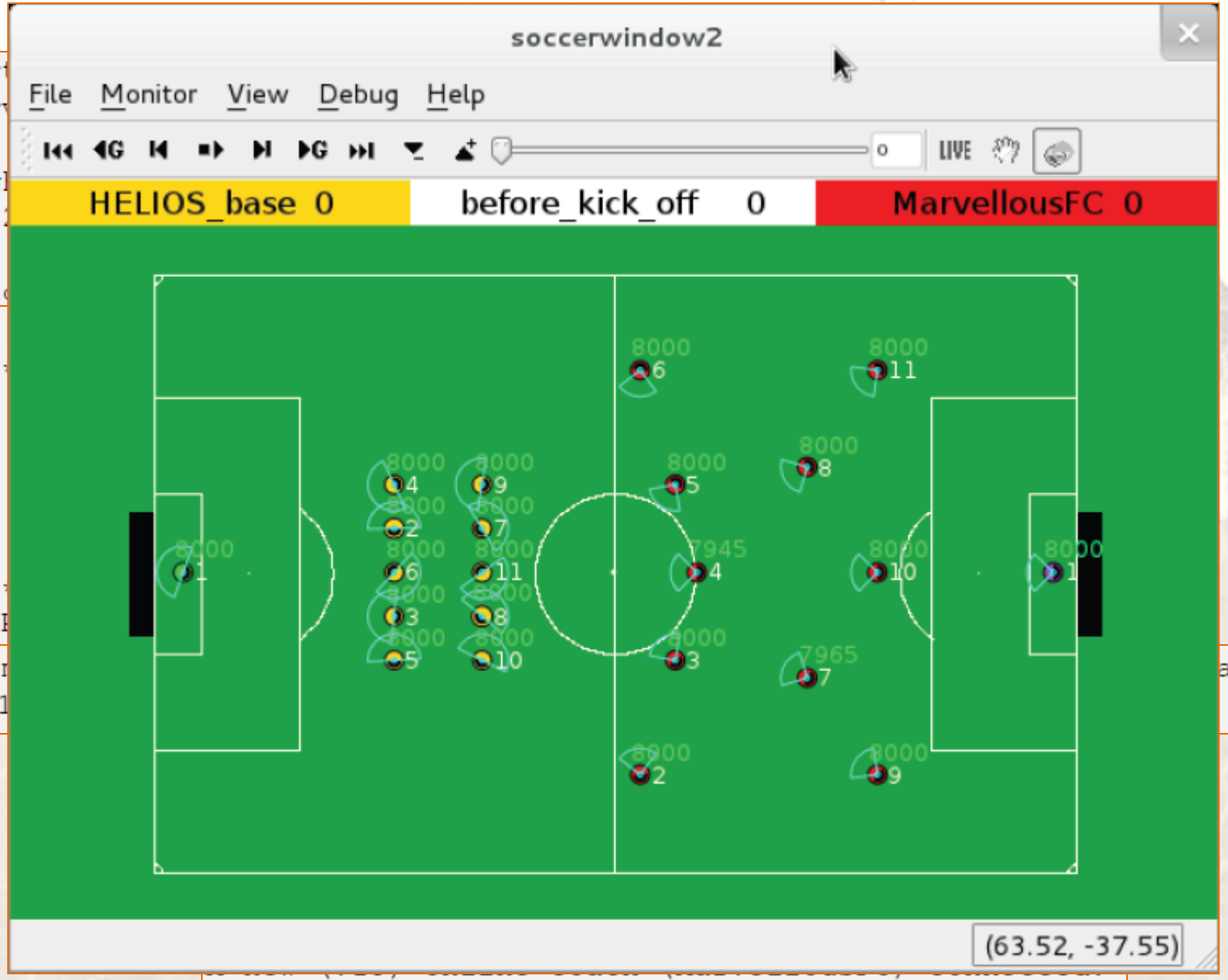
http://sourceforge.net/apps/mediawiki/sserver/index.php?title=RoboCup2011/Competition

Final Ranking

1. WrightEalge
2. HELIOS2011
3. MarliK
4. Oxsy
5. ESKILAS
6. NADCO-2D
7. FCPortugal
8. RaiC2011
9. HfutEngine2011
10. Apollo
11. aua2d
12. Edinferno.2D
13. Ri-one
14. Dainamite
15. AbouAliSina
16. ArtSapience
17. Photon

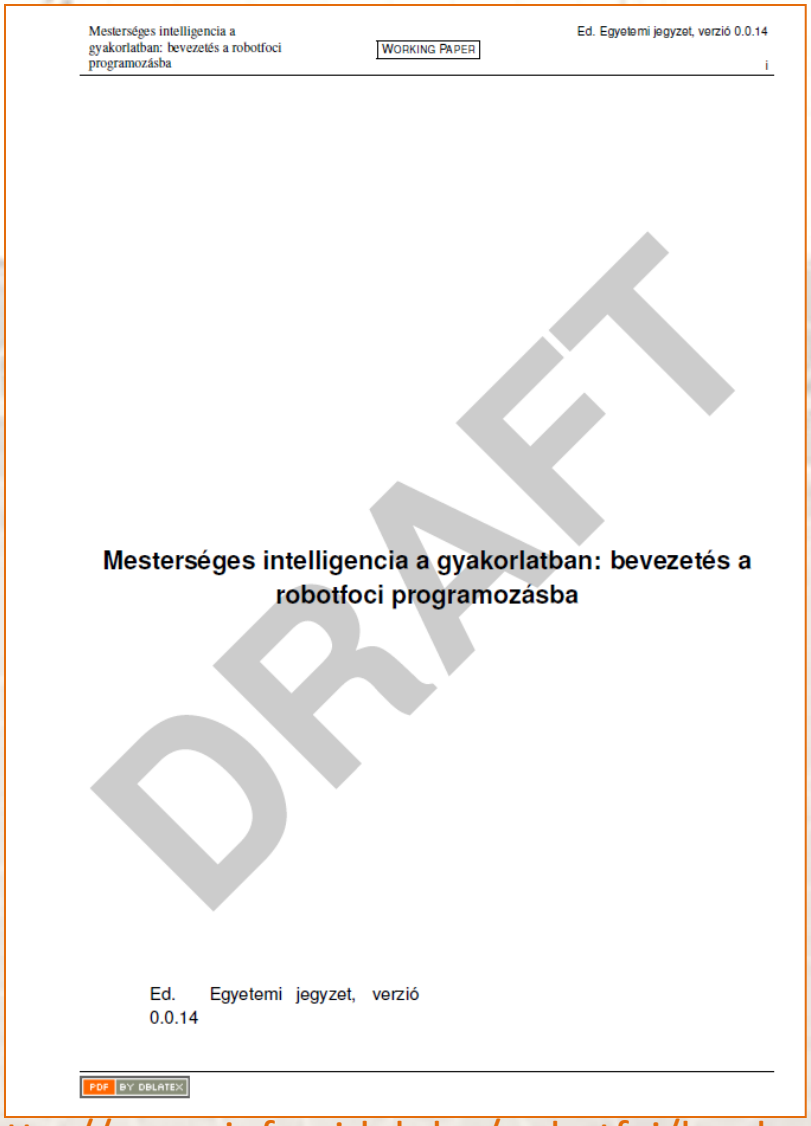
http://www.ustream.tv/recorded/15907824

A szoftverek használata



Ism: A robotfoci labortámogatása

Bátfai Norbert: ***Mesterséges intelligencia a gyakorlatban: bevezetés a robotfoci programozásba***



http://www.inf.unideb.hu/~nbatfai/book.pdf

Ism: BNF, Backus Normal Form (P1 ism.)

John Backus, ALGOL 60
Környezetfüggetlen nyelvekhez

<nem terminális> ::= konkatenációja terminálisoknak, nem terminálisoknak, illetve {iteráció}, [opcionális], alter|natíva

<egész szám> ::= <előjel><szam>
<előjel> ::= [-|+]
<szam> ::= <szamjegy>{<szamjegy>}
<szamjegy> ::= 0|1|2|3|4|5|6|7|8|9

RCSS protokollok

A kliens ágensek érzékelési protokollja

ObjInfo ::= (***ObjName Distance Direction DistChange DirChange BodyFacingDir HeadFacingDir***)
| (***ObjName Distance Direction DistChange DirChange***
| (***ObjName Distance Direction***)
| (***ObjName Direction***)

ObjName ::= (**p** [”*Teamname*” [*UniformNumber* [**goalie**]]])
| (**b**)
| (**g** [**l**|**r**])
| (**f c**)
| (**f** [**l**|**c**|**r**] [**t**|**b**])
| (**f p** [**l**|**r**] [**t**|**c**|**b**])
| (**f g** [**l**|**r**] [**t**|**b**])
| (**f** [**l**|**r**|**t**|**b**] **0**)
| (**f** [**t**|**b**] [**l**|**r**] [**10**|**20**|**30**|**40**|**50**])
| (**f** [**l**|**r**] [**t**|**b**] [**10**|**20**|**30**])
| (**l** [**l**|**r**|**t**|**b**])
| (**B**)
| (**F**)
| (**G**)
| (**P**)

Distance ::= positive real number
Direction ::= **-180** ∼**180** degrees
DistChange ::= real number
DirChange ::= real number
HeadFaceDir ::= **-180** ∼**180** degrees
BodyFaceDir ::= **-180** ∼**180** degrees
Teamname ::= string
UniformNumber ::= 1 ∼11

http://netcologne.dl.sourceforge.net/project/sserver/rcssmanual/9-20030211/manual-20030211.pdf
http://sourceforge.net/projects/sserver/files/rcssmanual/

RCSS protokollok, a látás érzékelés

(see *Time* *ObjInfo*$^+$)

Time ::= simulation cycle of the soccerserver

ObjInfo ::= (*ObjName Distance Direction DistChange DirChange BodyFacingDir HeadFacingDir*)
| (*ObjName Distance Direction DistChange DirChange*
| (*ObjName Distance Direction*)
| (*ObjName Direction*)

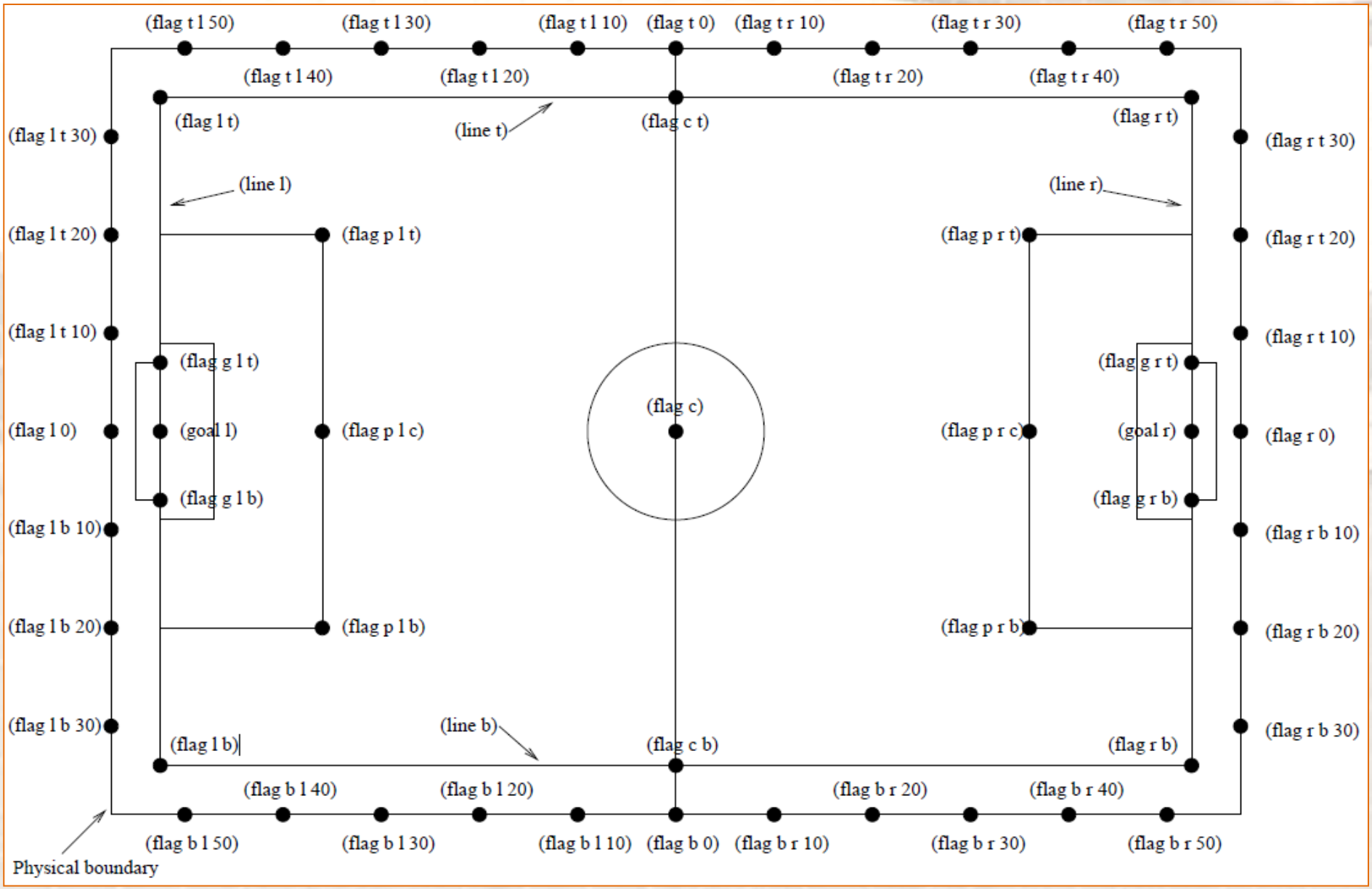
ObjName ::= (**p** ["*Teamname*" [*UniformNumber* [**goalie**]]])
| (**b**)
| (**g** [**l**|**r**])
| (**f c**)
| (**f** [**l**|**c**|**r**] [**t**|**b**])
| (**f p** [**l**|**r**] [**t**|**c**|**b**])
| (**f g** [**l**|**r**] [**t**|**b**])

```
(see 0 ((f c) 39.6 28 0 0) ((f c t) 38.1 -23 0 0) ((f r t) 89.1 -10) ((f r b) 102.5 31) ((f ↩
g r b) 90.9 17) ((g r) 89.1 12) ((f g r t) 88.2 8) ((f p r b) 81.5 29) ((f p r c) 73.7 ↩
15) ((f p r t) 70.8 -1) ((F) 1.5 -131) ((f t 0) 40.4 -30) ((f t r 10) 49.4 -24) ((f t r ↩
20) 58.6 -20) ((f t r 30) 68 -17) ((f t r 40) 77.5 -15) ((f t r 50) 87.4 -13) ((f t l ↩
10) 32.1 -39) ((f b r 30) 87.4 42) ((f b r 40) 94.6 38) ((f b r 50) 102.5 34) ((f r 0) ↩
94.6 12) ((f r t 10) 92.8 6) ((f r t 20) 92.8 -1) ((f r t 30) 92.8 -7) ((f r b 10) 96.5 ↩
17) ((f r b 20) 100.5 23) ((f r b 30) 104.6 28) ((b) 40.4 28) ((l r) 87.4 90))
```

Distance ::= positive real number
Direction ::= **-180** ∼**180** degrees
DistChange ::= real number
DirChange ::= real number
HeadFaceDir ::= **-180** ∼**180** degrees
BodyFaceDir ::= **-180** ∼**180** degrees
Teamname ::= string
UniformNumber ::= 1 ∼11

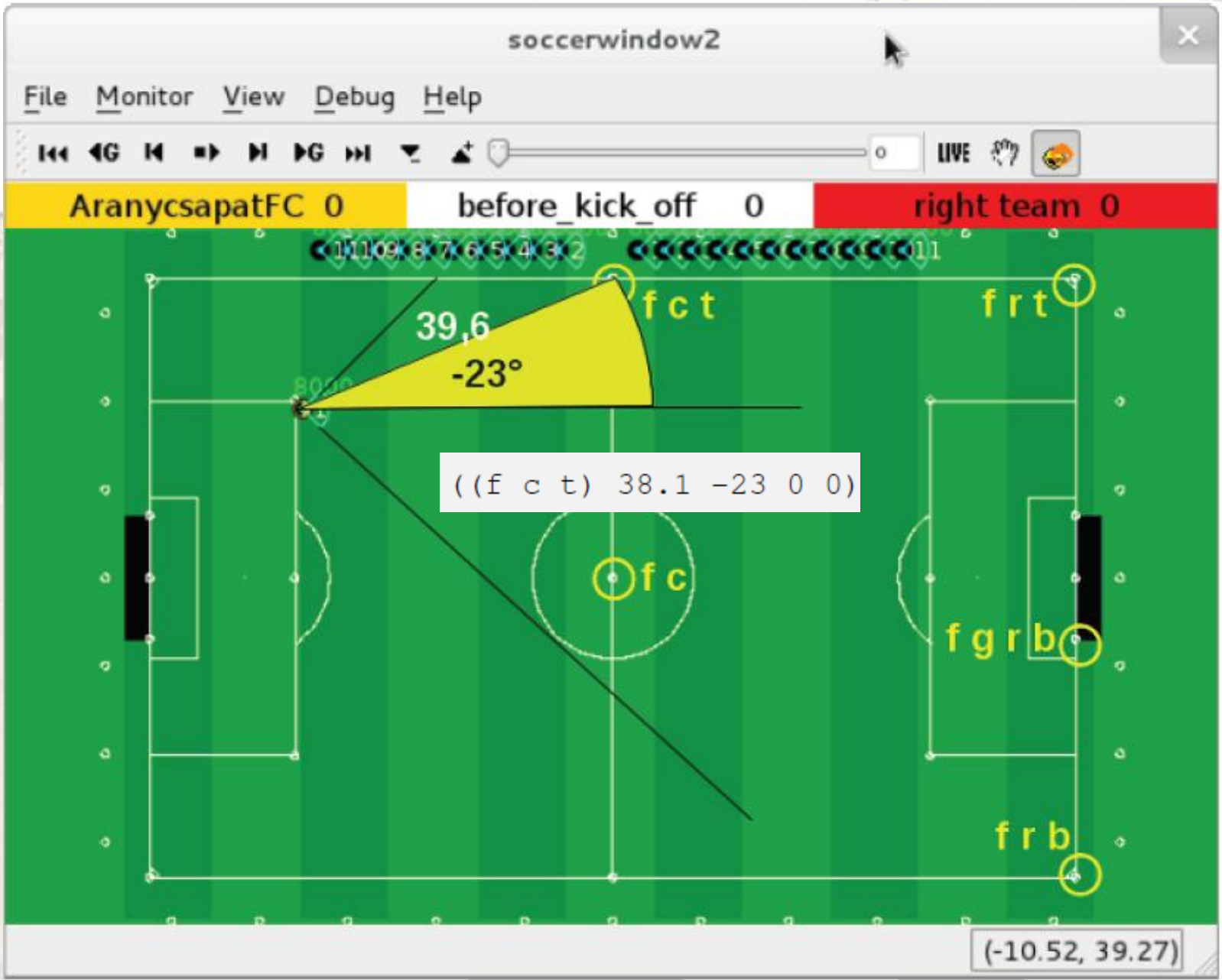
http://netcologne.dl.sourceforge.net/project/sserver/rcssmanual/9-20030211/manual-20030211.pdf
http://sourceforge.net/projects/sserver/files/rcssmanual/

Az 55 zászló



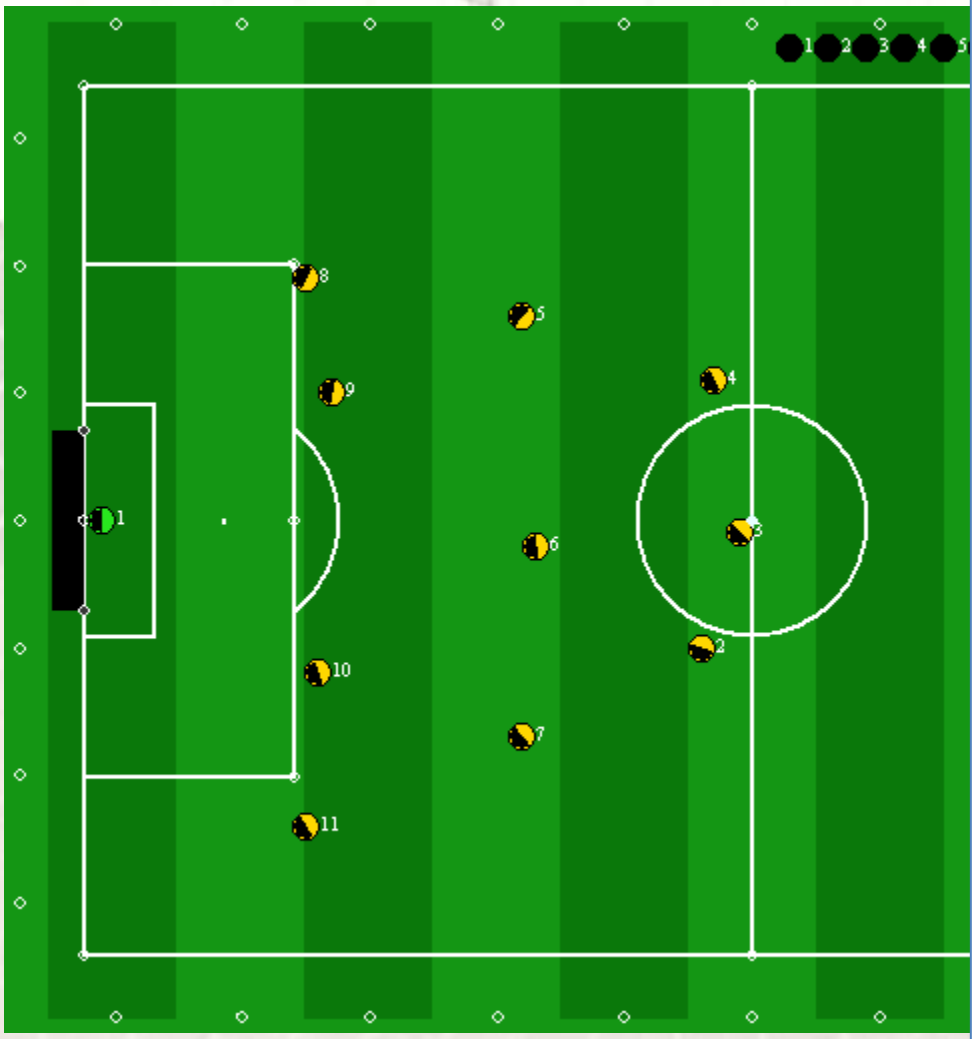
http://netcologne.dl.sourceforge.net/project/sserver/rcssmanual/9-20030211/manual-20030211.pdf
http://sourceforge.net/projects/sserver/files/rcssmanual/

A látás érzékelés



Ism: RCSS protokollok

A pálya

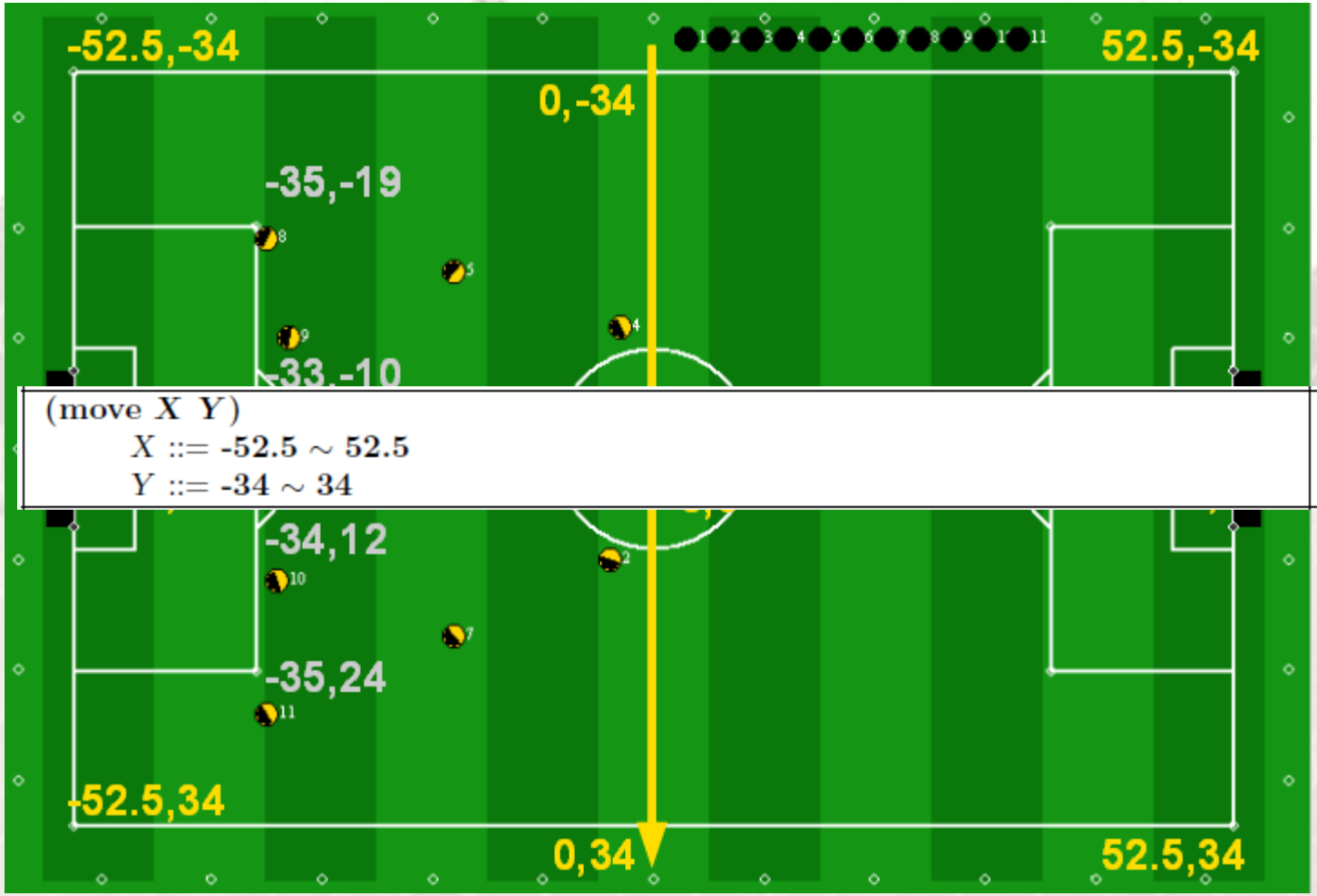


```
/* 4-3-3 */
@Override
protected void kozepkezdes() {

  switch (getPlayer().getNumber()) {

    // Vedo

    case 8:
      getPlayer().move(-35, -19);
      break;

    case 9:
      getPlayer().move(-33, -10);
      break;

    case 10:
      getPlayer().move(-34, 12);
      break;

    case 11:
      getPlayer().move(-35, 24);
      break;

  }

}
```

http://netcologne.dl.sourceforge.net/project/sserver/rcssmanual/9-20030211/manual-20030211.pdf
http://sourceforge.net/projects/sserver/files/rcssmanual/

Ism: RCSS protokollok

A pálya

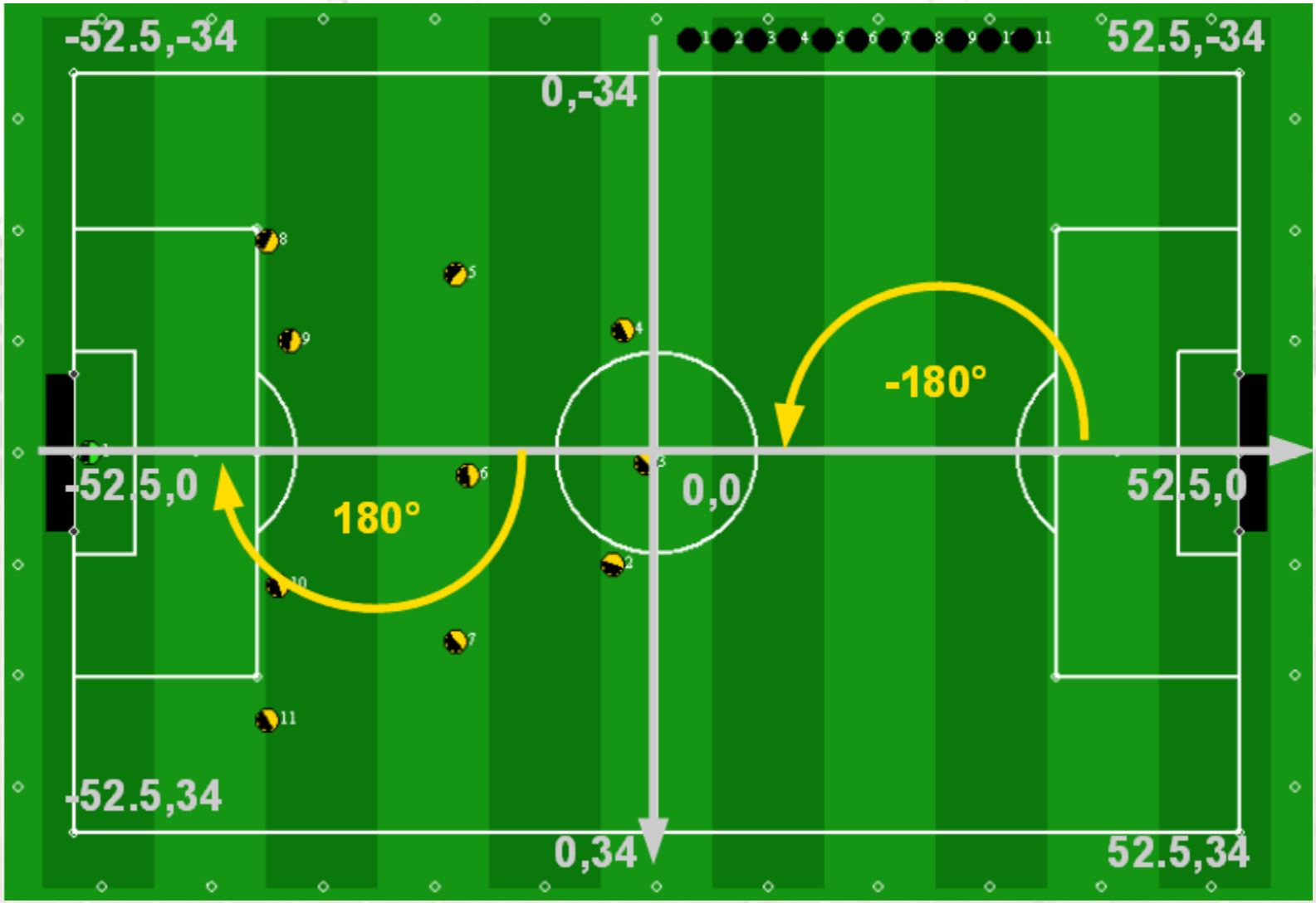


(move X Y) X ::= -52.5 $\sim$ 52.5 Y ::= -34 $\sim$ 34	Yes

http://netcologne.dl.sourceforge.net/project/sserver/rcssmanual/9-20030211/manual-20030211.pdf
http://sourceforge.net/projects/sserver/files/rcssmanual/

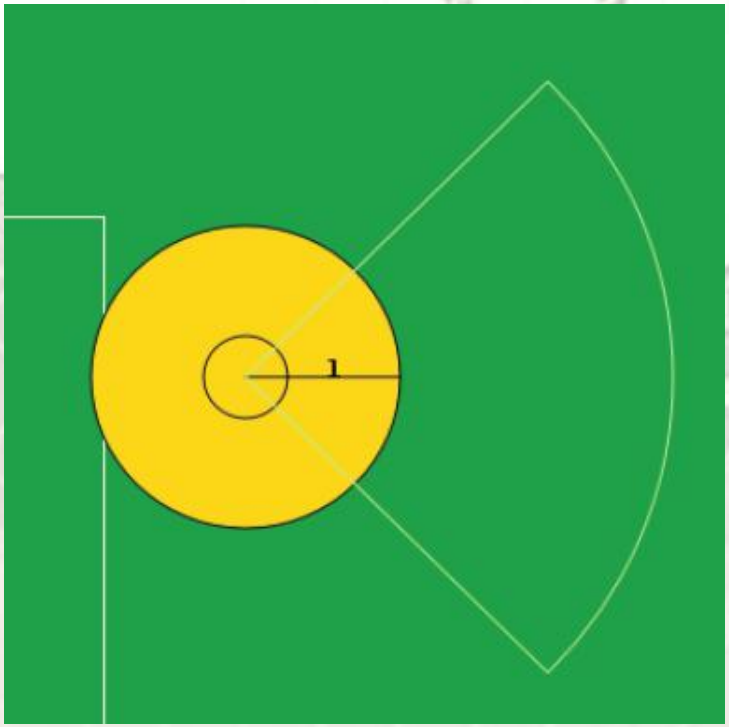
Ism: RCSS protokollok

A szögek értelmezése a pályán

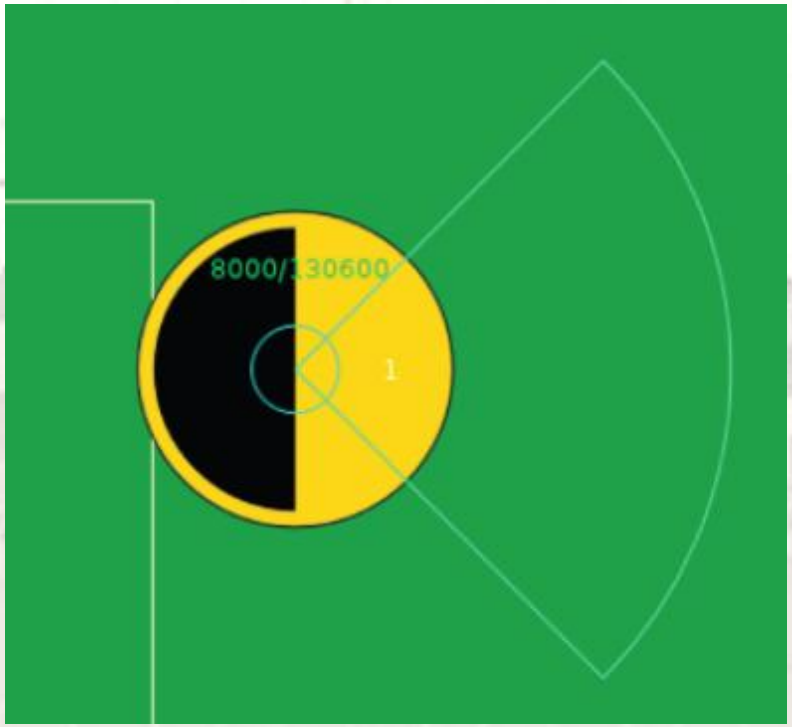


http://netcologne.dl.sourceforge.net/project/sserver/rcssmanual/9-20030211/manual-20030211.pdf
http://sourceforge.net/projects/sserver/files/rcssmanual/

Mozgás a pályán

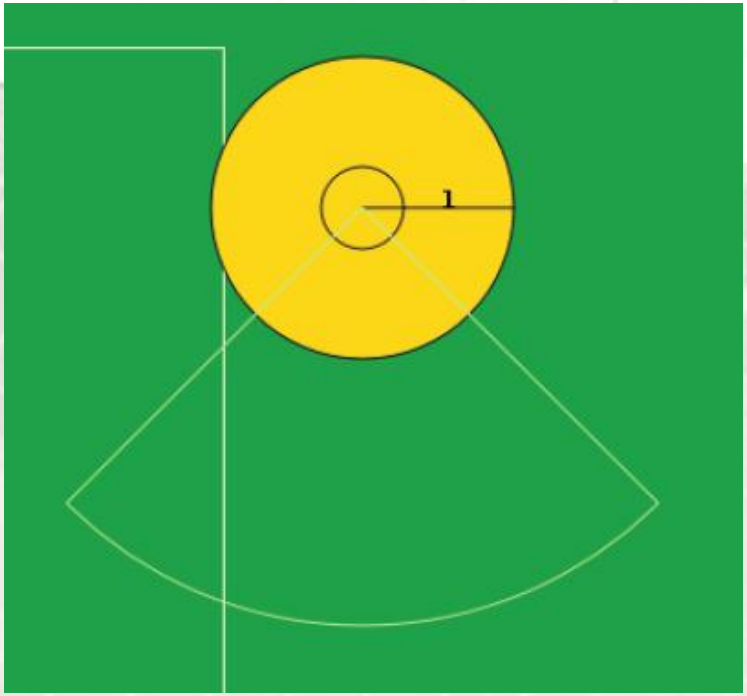


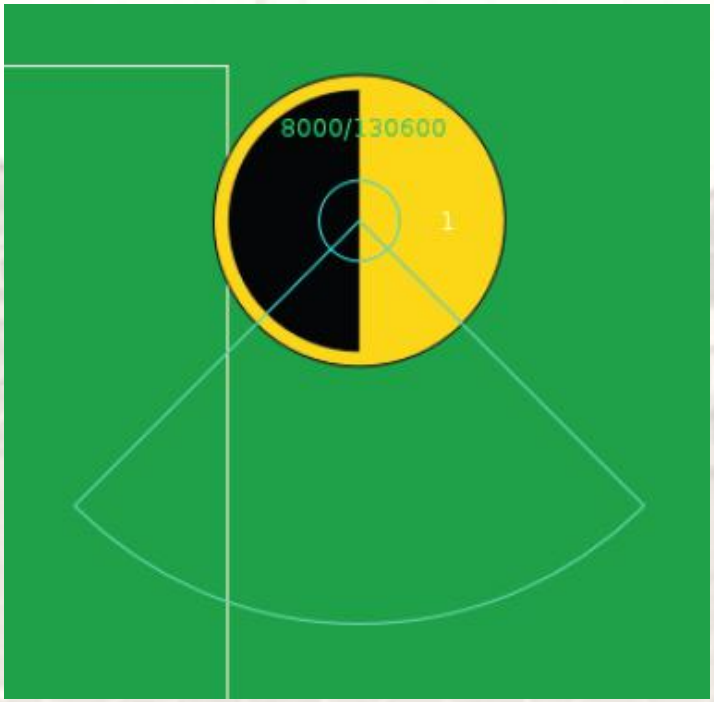
rcssmonitor



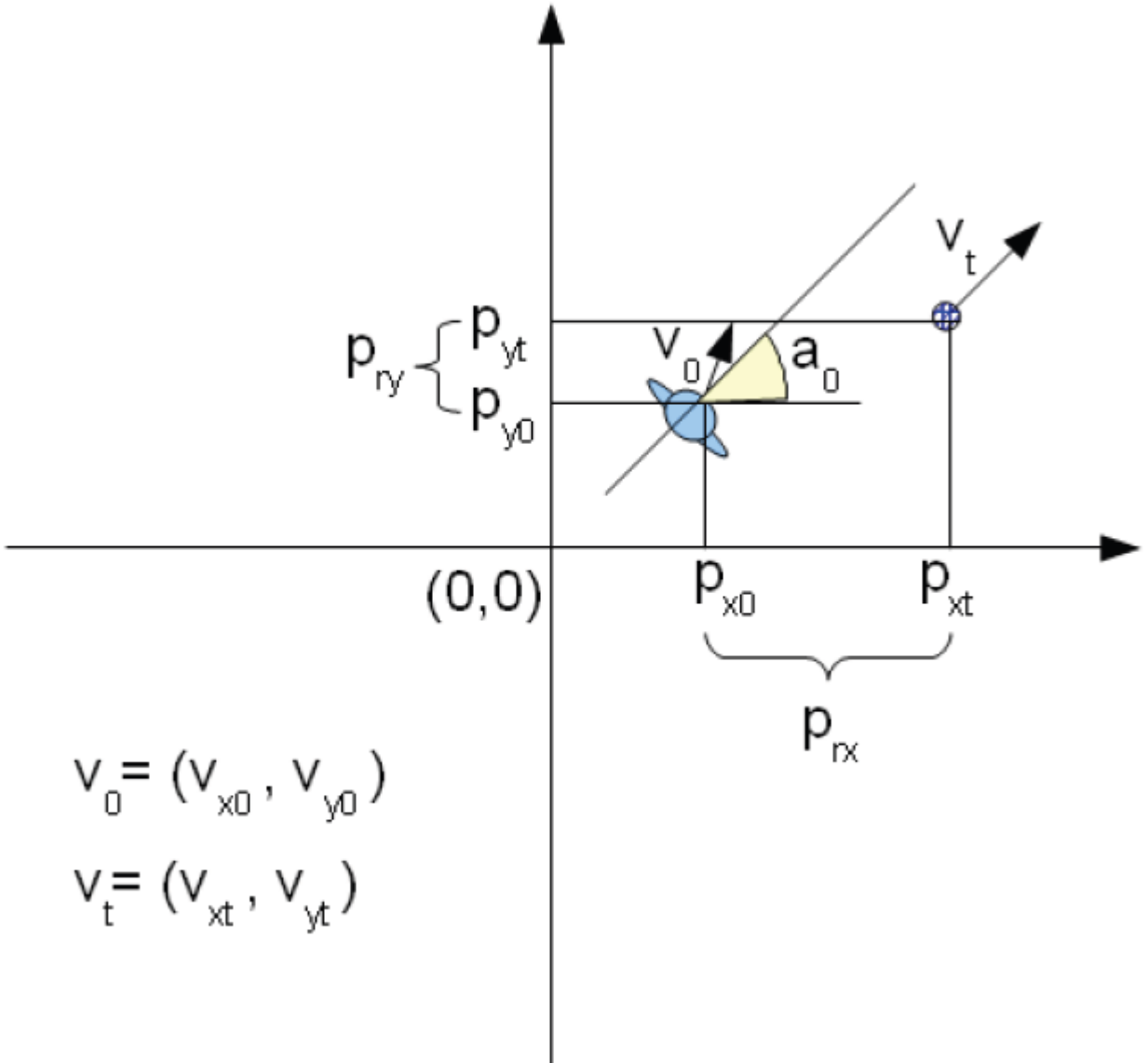
soccerwindow2

Mozgás a pályán





Mozgás a pályán

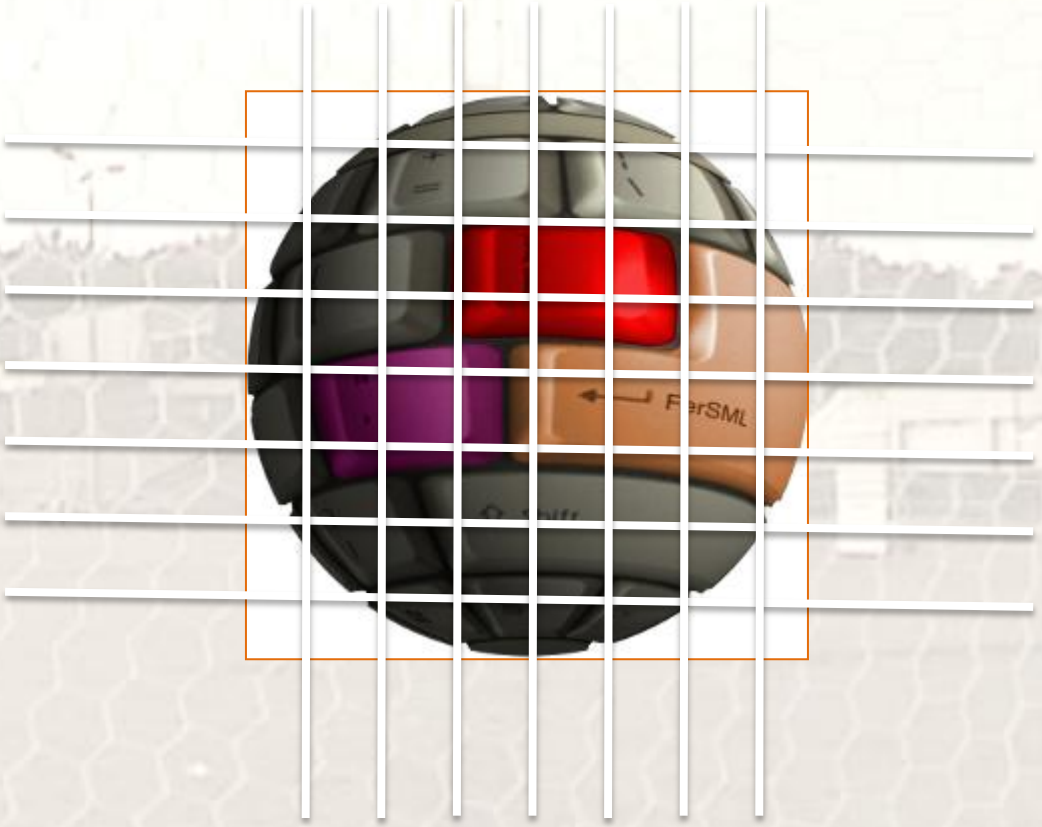


Mozgás a pályán

• A (**move** x_koordináta y_koordináta) parancs: paramétereiben megadott koordinátára állítja a játékost, de csakis középkezdéskor, azaz a félidők elején és a gólok után van hatása, szimulációs ciklusonként egyszer adható ki. A középkezdés felállásának megadásakor azt tételezzük fel, hogy a szóban forgó csapat a bal oldalon áll, azaz az x koordináták mindenképpen negatívak! Például: (move -35 -19).
• A (**dash** -100_tól_+100_ig_az_erő) parancs: a játékost az adott erővel meglöki abban az irányban, amiben a játékos teste áll, szimulációs ciklusonként egyszer adható ki. A játékos állóképessége a megadott erővel, ha annak előjele negatív, akkor annak kétszeresével csökken. Fontos látni, hogy adott esetben a játékos testének iránya és sebességvektora (az ábrán a v0) eltérő irányú lehet (például éppen ciklusokon át mozog a játékos, amikor közben kap egy turn majd egy dash parancsot a következő ciklusban). Például: (dash 40).
• A (**turn** -180_tól_+180_ig_a_szög) parancs: elfordítja a játékos testét. A szög a test aktuállis álláshoz relatív.
• A (**turn_neck** -180_tól_+180_ig_a_szög) parancs: a játékos testétől függetlenül, ahhoz relatívan (és maximum -90, +90) tartományban elforgatja a fejét. Fontos, hogy ezzel (és nem csak a turn paranccsal, ami ugye a testtel együtt nyilván a fejet is fordítja) egyetemben a játékos látószöge is változik. Szimulációs ciklusonként egyszer adható ki, de lehet együtt hívni a turn, move vagy kick parancsokkal. Például: (turn 15).
• A (**catch** -180_tól_+180_ig_a_szög) parancs: ez egy kapus parancs, (a kapus testéhez relatív) adott irányban megpróbálja elkapni a labdát.

 = http://www.inf.unideb.hu/~nbatfai/book.pdf , a további parancsokat lásd a könyvben!

FerSML platform logo



(Ismétlő) laborkártyák

```
nbatfai@hallg:~$ g++ maini.cpp int.cpp -o binszamfa
nbatfai@hallg:~$ ./binszamfa
55 66 77 33 11 22 99 11 66 60
------------99(1, 3)
---------77(1, 2)
------66(2, 1)
---------60(1, 2)
---55(1, 0)
------33(1, 1)
------------22(1, 3)
---------11(2, 2)
```

```
int Int::compare(Int& n) {
    if (ertek == n.ertek)
        return 0;
    else if (ertek > n.ertek)
        return 1;
    else
        return -1;
}
```

Mi szükség erre a függvényre a bináris fát absztraháló osztályban?

Laborkártyák

```
#ifndef INT__H
#define INT__H

#include <iostream>

class
{
public
    In
    ~I

    In
    }
    Int & operator= (const Int &n);

    int compare(Int& n);
    Int operator+(const Int& n) const ;
    Int& operator+=(const Int& n);

    friend std::ostream& operator<< (std::ostream& os, const Int& n);
    friend std::istream& operator>> (std::istream& is, Int& n);

private:
    int ertek;
};

#endif
```

```
#include <iostream>

class Int
{
public:
    Int (int n=0):ertek (n) {std::cout << "-- Int ctor " << &ertek << std::endl << std::flush;}
    ~Int () {std::cout << "-- Int dtor " << &ertek << std::endl << std::flush;};

    Int (const Int &n) {
        ertek = n.ertek;
        std::cout << "-- Int masolo ctor " << &ertek << std::endl << std::flush;
    }
```

Laborkártyák

```
#include "int.h"

Int& Int::operator= (const Int &n) {
    ertek = n.ertek;
    return *this;
}
int Int::compare(Int& n) {
    if (ertek == n.ertek)
        return 0;
    else if (ertek > n.ertek)
        return 1;
    else
        return -1;
}

Int
Int::operator+(const
    return Int(ertek
}

Int&
Int::operator+=(const
    ertek += n.ertek;
    return *this;
}

std::ostream&
operator<< (std::ostream& os, const Int& n) {
    os << n.ertek;
    return os;
}
```

```
#include "int.h"

Int& Int::operator= (const Int &n) {
    std::cout << "-- Int masolo ertekadas "  << &ertek << std::endl << std::flush;
    ertek = n.ertek;
    return *this;
}

Int
Int::operator+(const Int& n) const {
    std::cout << "-- Int + tulterheles "  << &ertek << std::endl << std::flush;
    return Int(ertek + n.ertek);
}
/*
Int
Int::operator+(const Int& n) const {
    std::cout << "-- Int + tulterheles "  << &ertek << std::endl << std::flush;
    Int osszeg(ertek + n.ertek);
    return osszeg;
}
*/
```

Laborkártyák

```
#include <iostream>
#include "int.h"

int
main () {

    std::cout << "I
    Int a=40, b=2;

    std::cout << "s
    std::cout << b+

    std::cout << "I
    Int * ip = new

    std::cout << "a
    a = b;

    std::cout << "I
    Int c = a;

    std::cout << "c
    delete ip;

    std::cout << "r
    return 0;
}
```

mikor : bash

Fájl Szerkesztés Nézet Könyvjelzők Beállítások Súgó

```
[norbi@sgu mikor]$ g++ main.cpp int.cpp -o mikormi
[norbi@sgu mikor]$ ./mikormi
Int a=40, b=2;
-- Int ctor 0x7fff0342af50
-- Int ctor 0x7fff0342af40
std::cout << b+a << std::endl;
-- Int + tulterheles 0x7fff0342af40
-- Int ctor 0x7fff0342af60
42
-- Int dtor 0x7fff0342af60
Int * ip = new Int(42);
-- Int ctor 0x123e010
a = b;
-- Int masolo ertekadas 0x7fff0342af50
Int c = a;
-- Int masolo ctor 0x7fff0342af30
delete ip;
-- Int dtor 0x123e010
return 0;
-- Int dtor 0x7fff0342af30
-- Int dtor 0x7fff0342af40
-- Int dtor 0x7fff0342af50
[norbi@sgu mikor]$
```

mikor : bash

Laborkártya kérdések

Mi történik (mi: például másoló értékadás, kostruktor stb. és milyen sorrendben fut le) ha az alábbi módosításokat tesszük:

```
#include <iostream>
#include "int.h"

int
main () {

    std::cout << "Int a=40, b=2;" << std::endl;
    Int a=40, b=2;

    std::cout << "std::cout << b+a << std::endl;" << std::endl;
    std::cout << b+a << std::endl;

    std::cout << "Int * ip = new Int(42);" << std::endl;
    Int * ip = new Int(42);

    std::cout << "a = b;" << std::endl;
    a = b;

    std::cout << "Int c = a;" << std::endl;
    Int c = a;

    std::cout << "delete ip;" << std::endl;
    delete ip;

    std::cout << "return 0;" << std::endl;
    return 0;
}
```

Int e=a=b;

(b=b+a)

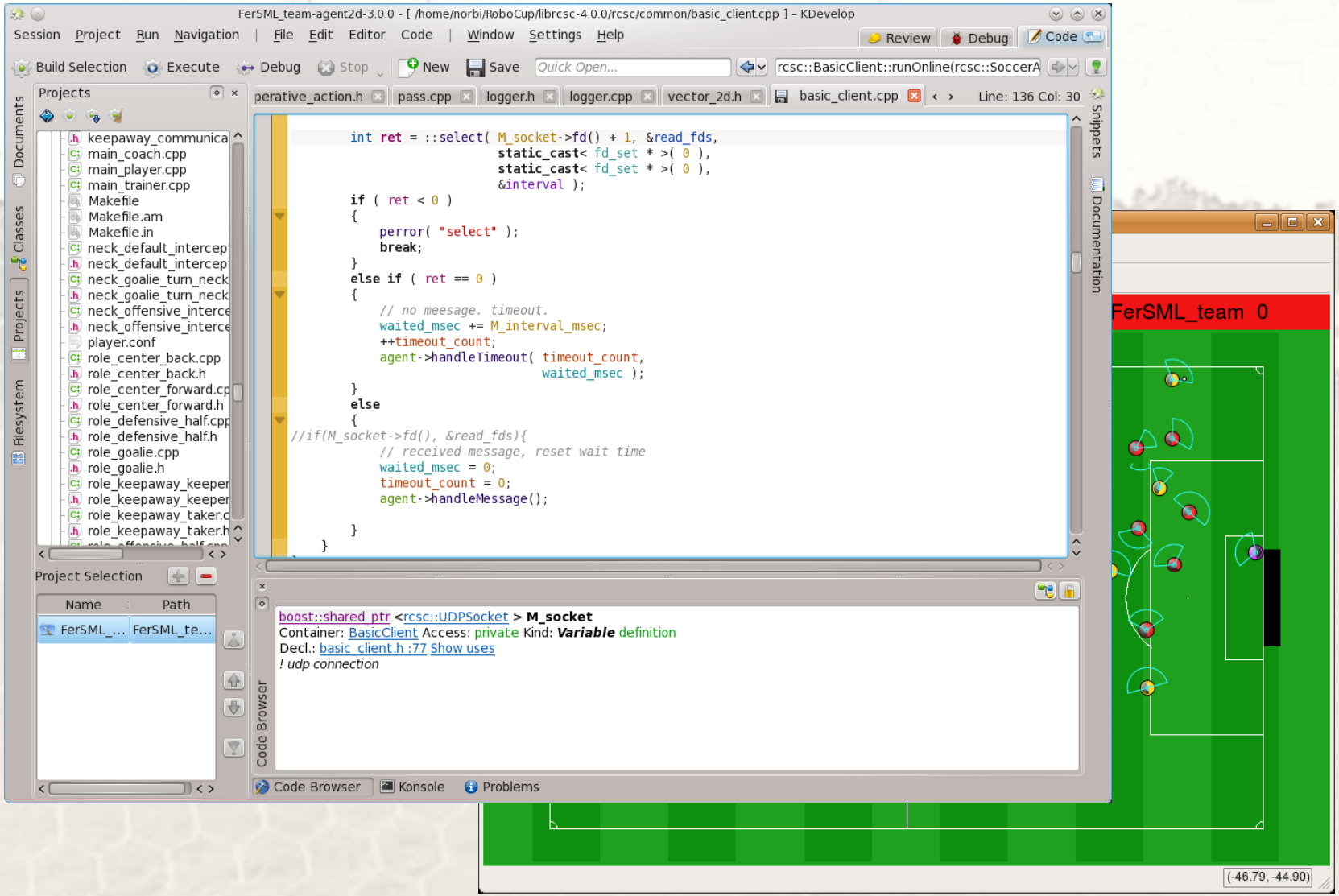
Int c = a+b;

a= a+b;

Int c = a+b;

Otthoni opcionális feladat

A robotfoci japán szoftvereinek (librcsc, agent2d) tanulmányozása a KDevelop-ban.



Otthoni opcionális feladat

A robotfoci japán szoftvereinek (librcsc, agent2d) tanulmányozása a KDevelop-ban.

Készítsd el az Arancsapat felállását a saját csapatodhoz!

WM (esetünkben 3-2-2-3), a „6-3”
4-2-4

Kötelező olvasmány

(B&L könyv)
Benedek Zoltán, Levendovszky Tihamér: Szoftverfejlesztés C++ nyelven, Budapest, 2007, Szak K.

218-244

(S könyv)
Stroustrup, Bjarne: A C++ programozási nyelv, Kiskapu, 2001.
http://webpac.lib.unideb.hu/WebPac/CorvinaWeb?action=onelong&showtype=longlong&recnum=255516&pos=3

431-452

„Nem tudok kimerítő leírást adni arról, hogy hogyan tudsz megtanulni programozni – nagyon összetett tudásról van szó. Egyet azonban elárulhatok: a könyvek és tanfolyamok nem érnek túl sokat (sok, valószínűleg a legtöbb hacker autodidakta). Aminek van értelme: (a) kódot olvasni és (b) kódot írni." - Eric Steven Raymond: How To Become A Hacker

A magyar fordítás: http://esr.fsf.hu/hacker-howto.html

Ajánlott olvasmány

(OSS könyv)
Alan Ezust, Paul Ezust: ***An Introduction to Design Patterns in C++ with Qt 4***, Prentice Hall (Open Source Series) 2006

Pdf-ben:
http://ptgmedia.pearsoncmg.com/images/9780131879058/downloads/0131879057_Ezust_book.pdf

214-219

(24 könyv)
Jesse Liberty, Horvath, David B. Büki András: ***Tanuljuk meg a C++ programozási nyelvet 24 óra alatt***
http://webpac.lib.unideb.hu/WebPac/CorvinaWeb?action=onelong&showtype=longlong&recnum=469876&pos=2

465-480

„Nem tudok kimerítő leírást adni arról, hogy hogyan tudsz megtanulni programozni – nagyon összetett tudásról van szó. Egyet azonban elárulhatok: a könyvek és tanfolyamok nem érnek túl sokat (sok, valószínűleg a legtöbb hacker autodidakta). Aminek van értelme: (a) kódot olvasni és (b) kódot írni." - Eric Steven Raymond: How To Become A Hacker

A magyar fordítás: http://esr.fsf.hu/hacker-howto.html

Ajánlott olvasmány

(CORBA könyv)
Csizmazia Balázs: CORBA-alapú elosztott alkalmazások
http://mek.niif.hu/01400/01404/01404.pdf

5-30 mazsolázgatva

„Nem tudok kimerítő leírást adni arról, hogy hogyan tudsz megtanulni programozni – nagyon összetett tudásról van szó. Egyet azonban elárulhatok: a könyvek és tanfolyamok nem érnek túl sokat (sok, valószínűleg a legtöbb hacker autodidakta). Aminek van értelme: (a) kódot olvasni és (b) kódot írni." - Eric Steven Raymond: How To Become A Hacker

A magyar fordítás: http://esr.fsf.hu/hacker-howto.html