Machine Learning wcielony (SER VII)

programowanie robotów LEGO z ev3dev.R

Bartosz Meglicki

IPI PAN, NCBJ

08\12\2014

SERATRON

Rysunek: SERATRON ;-)

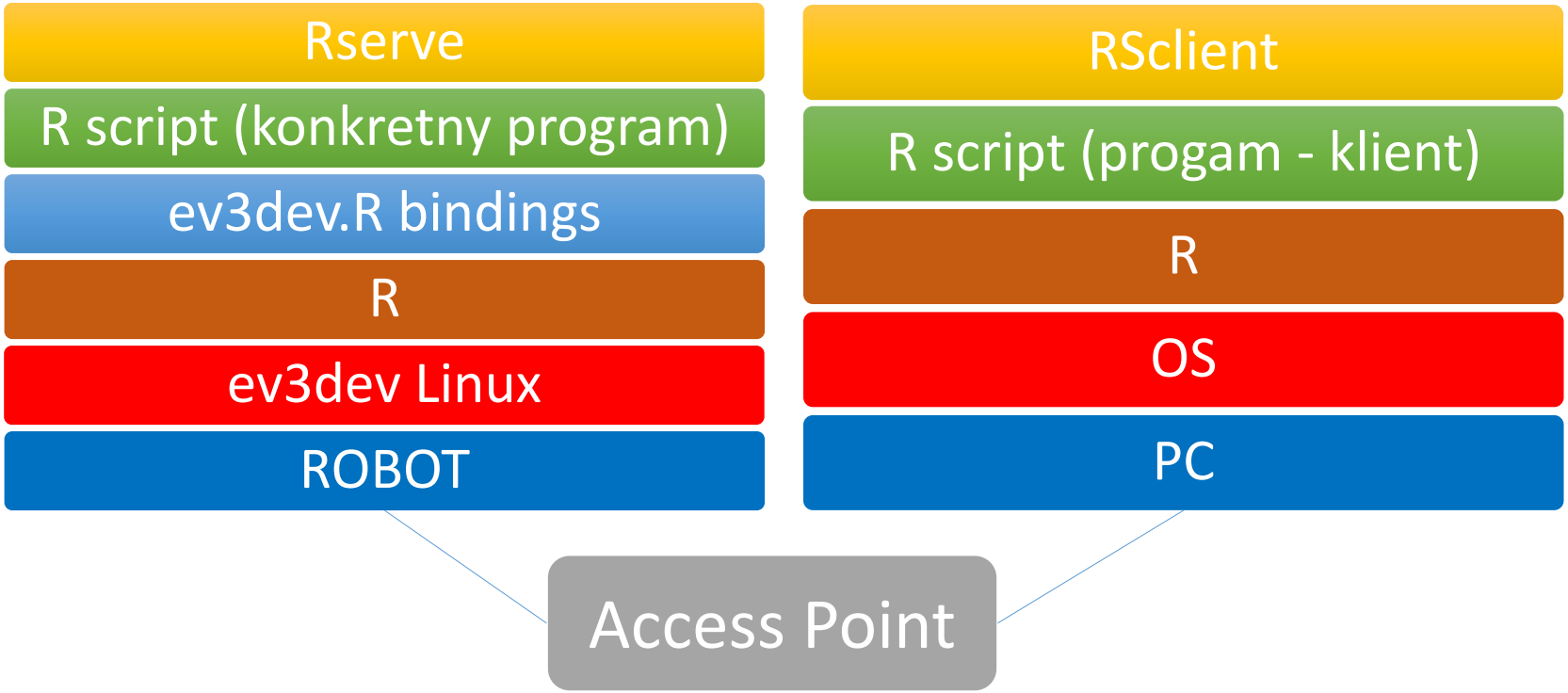
Głęboka woda...

```
...
 while( ( button <- Value(infrared) ) != 9 ) # Beacon
  {
    sleft=switch(button+1, 0L,  300L, -300L, 0L, 0L,
      300L, 300L, -300L, -300L, 0L, 0L, 0L)
    sright=switch(button+1, 0L,  0L, 0L, 300L, -300L,
      300L, -300L, 300L, -300L, 0L, 0L, 0L)

    SetPulsesPerSecondSP(left, sleft)
    SetPulsesPerSecondSP(right, sright)

    Sys.sleep(0.1)
  }
...
```

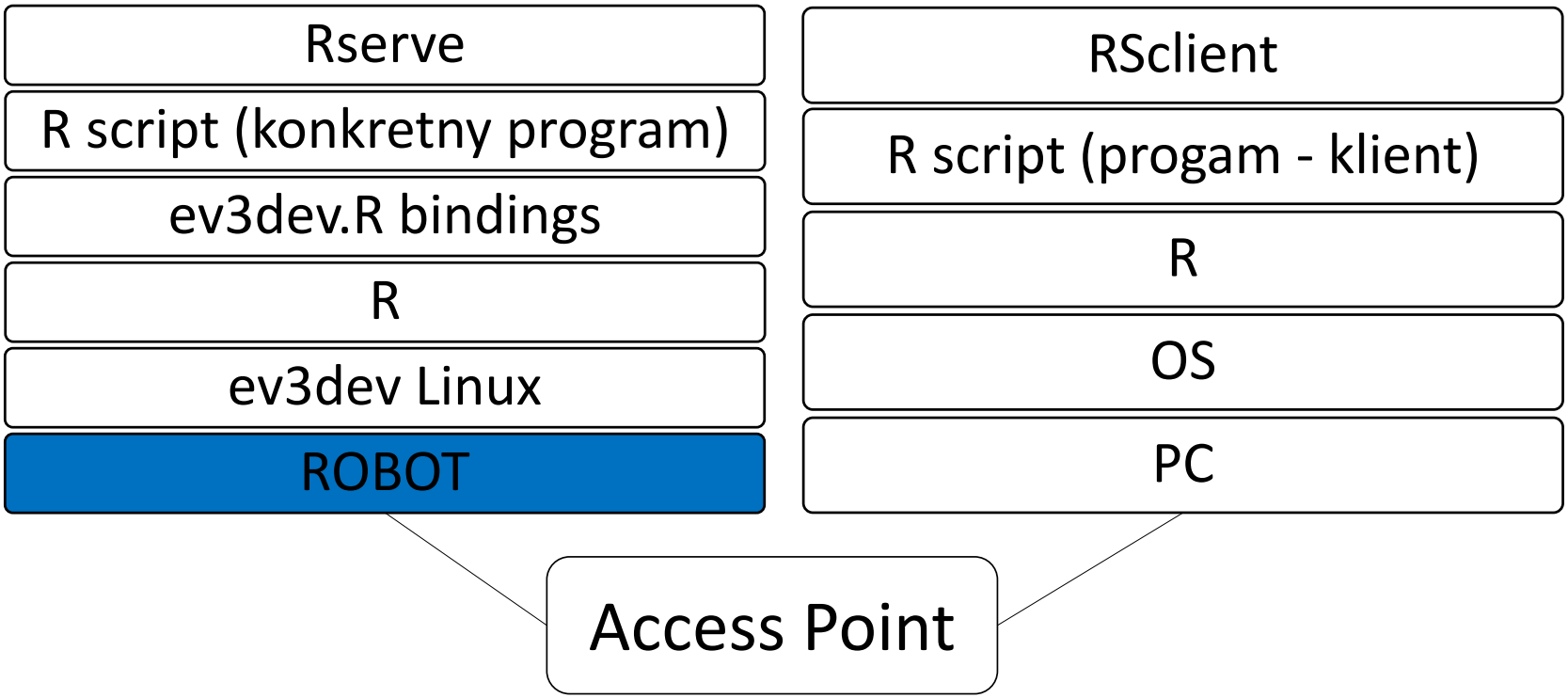
O autorze

- Wykształcenie
  - PW MiNI, Informatyka
    - inż.
    - mgr (MSI)
  - IPIPAN (w trakcie)
- Praca
  - NCBJ
  - Konstruktor/Programista
  - m.in. robotyka/automatyka
- R
  - IPI
  - NCBJ

W skrócie


Rserve
R script (konkretny program)
ev3dev.R bindings
R
ev3dev Linux
ROBOT
RSclient
R script (progam - klient)
R
OS
PC
Access Point

ROBOT

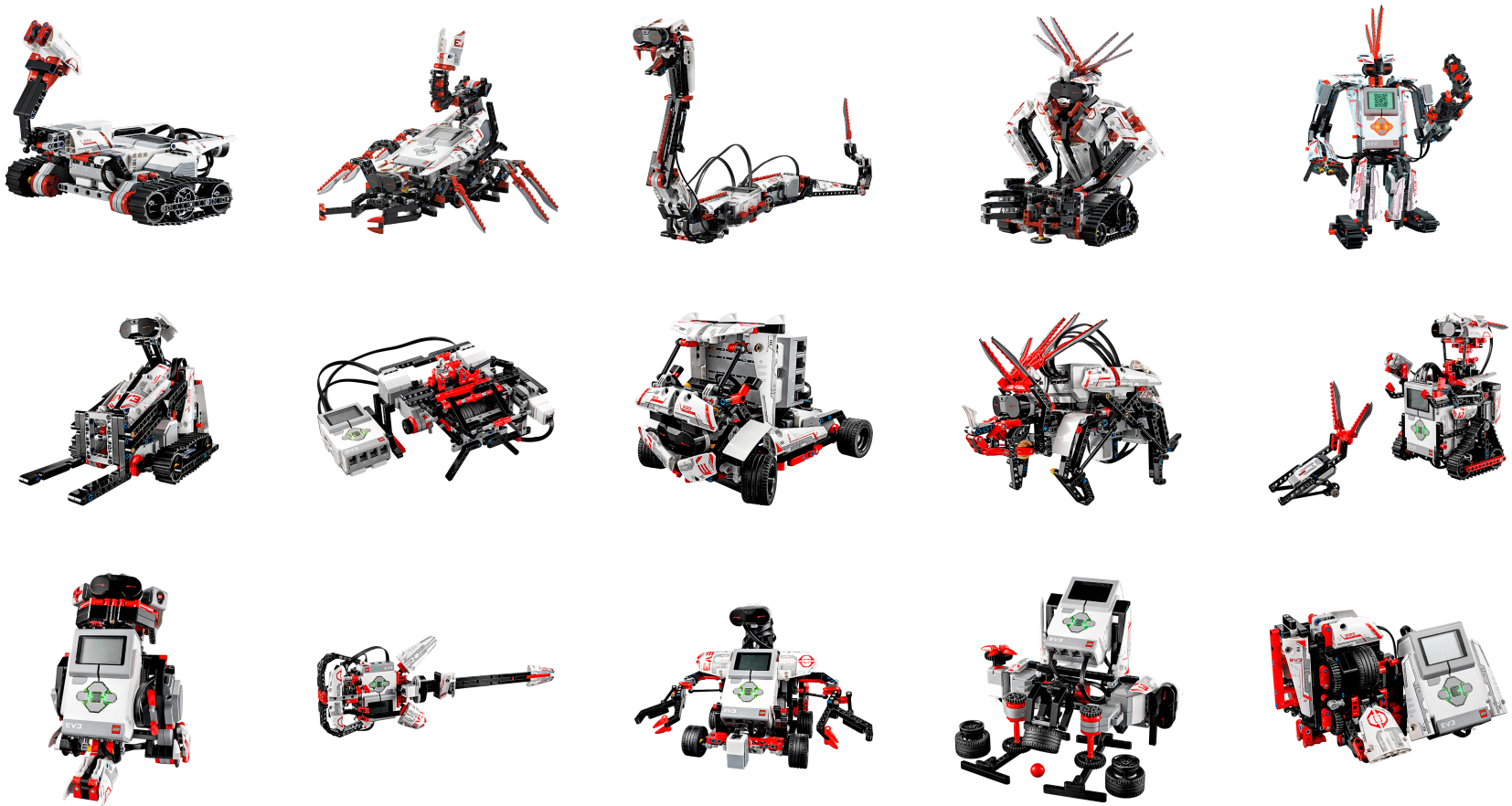

Rserve
R script (konkretny program)
ev3dev.R bindings
R
ev3dev Linux
ROBOT
RSclient
R script (progam - klient)
R
OS
PC
Access Point

Hardware

- LEGO Mindstorms EV3
- `http://mindstorms.lego.com`
- Od 09.2013

Hardware - przykładowe roboty

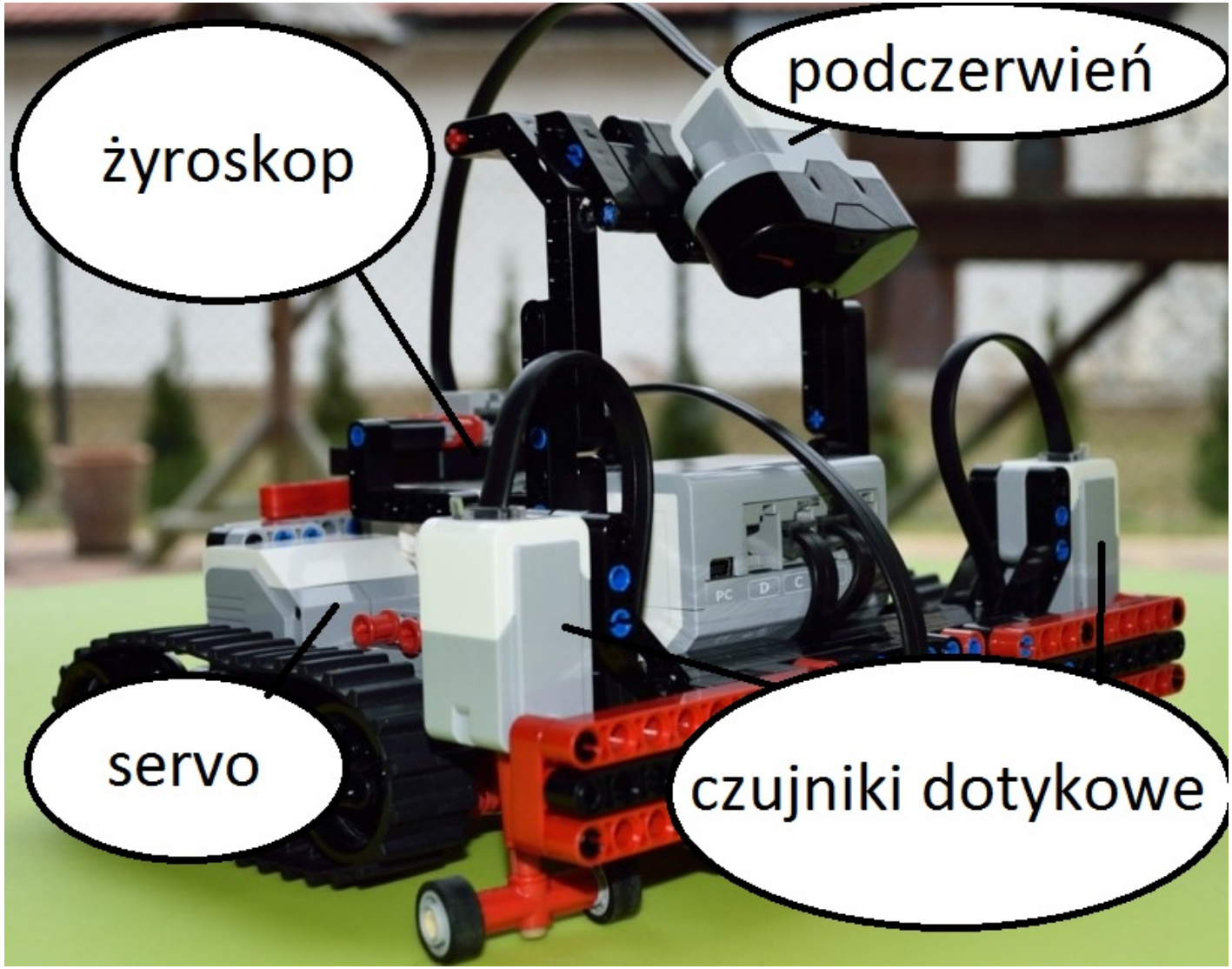
Rysunek: Przykladowe instrukcje (Home Edition)



Hardware - uwagi

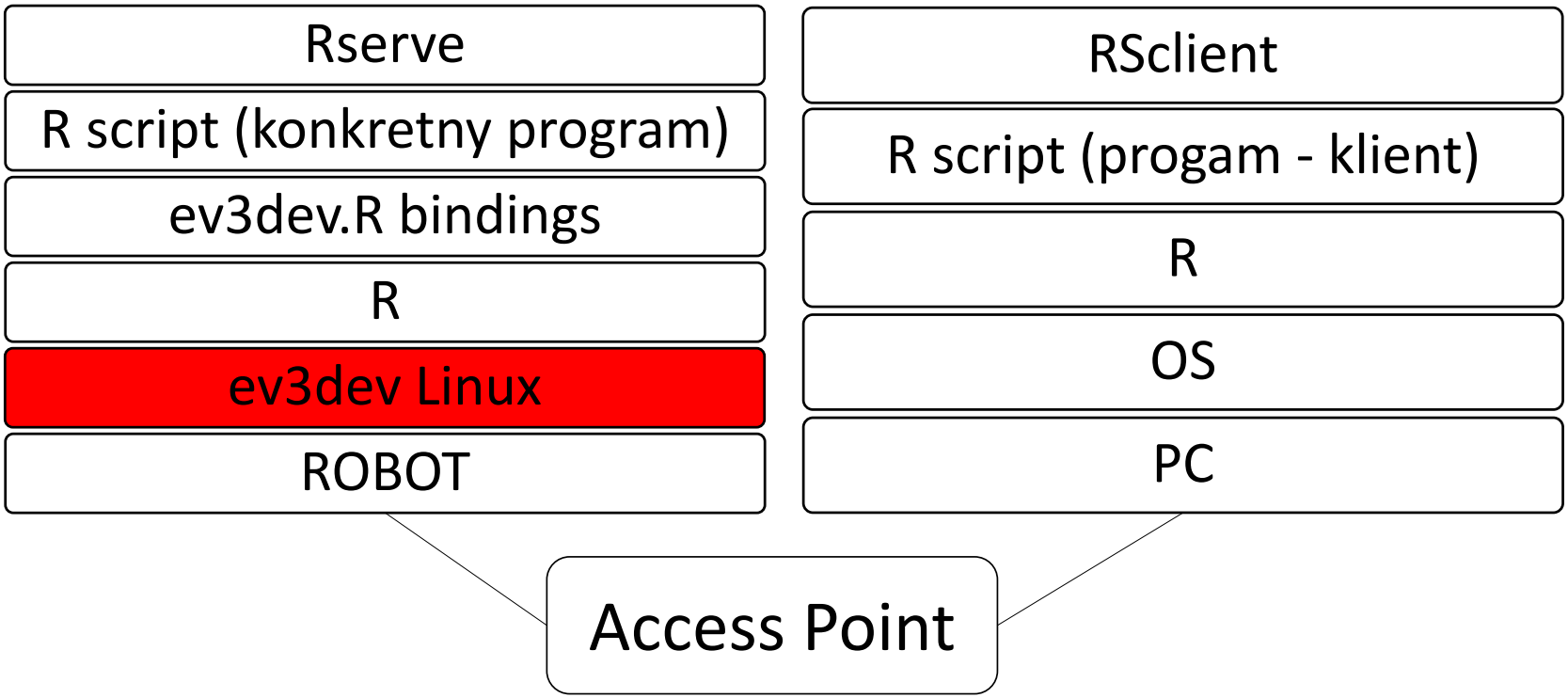
- 64 MB RAM (R!)
- SDHC standard, 2-32 GB
- Procesor AM1808 - nie ma FPU
- USB 1.1 Host - tylko 12 MBit/s
- ARM

SERATRON - hardware



Rysunek: SERATRON HW

ev3dev


Rserve
R script (konkretny program)
ev3dev.R bindings
R
ev3dev Linux
ROBOT
RSclient
R script (progam - klient)
R
OS
PC
Access Point

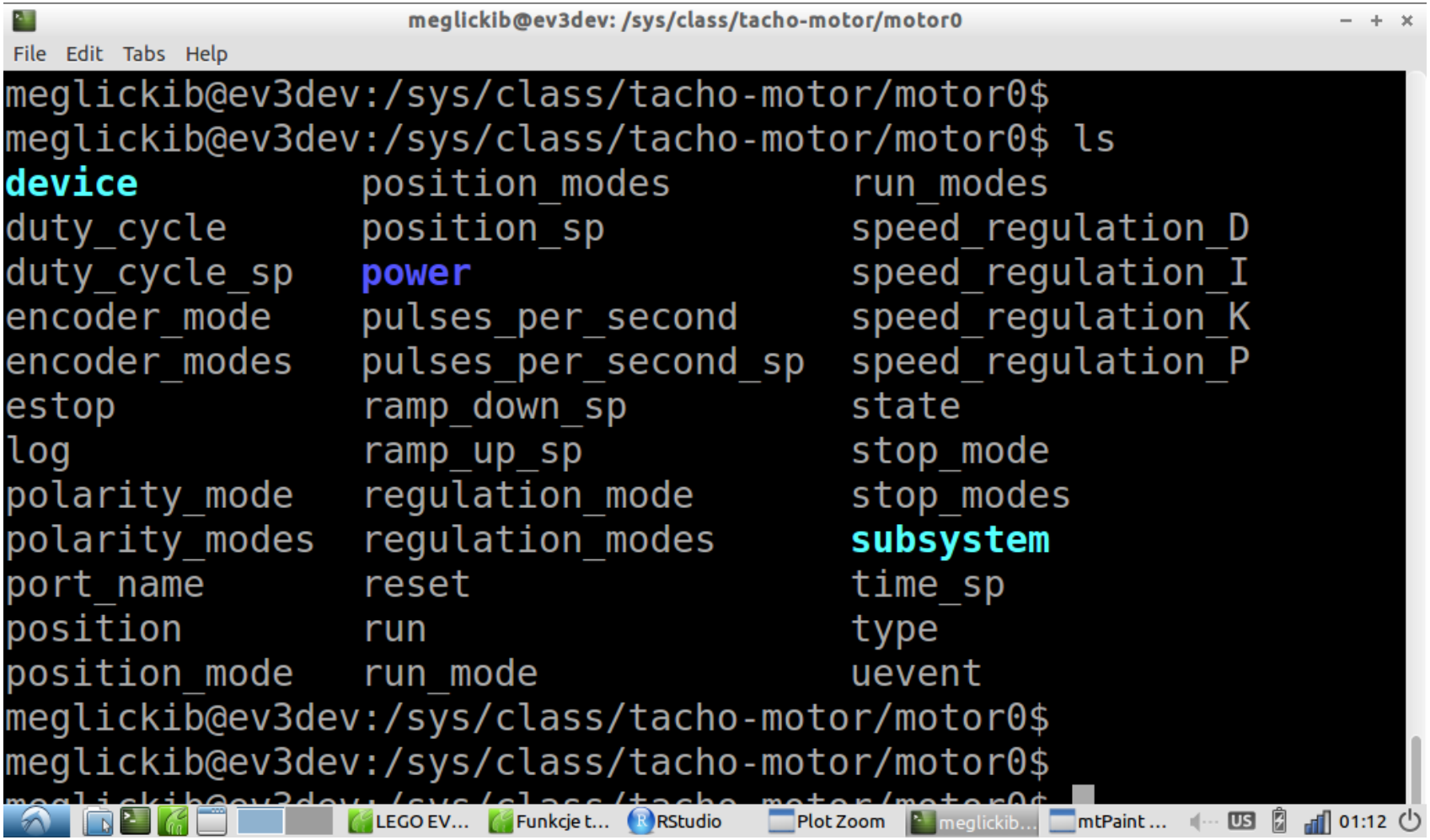
ev3dev

- Debian (jessie) Linux distro
- `http://www.ev3dev.org/`
- Dedykowany LEGO EV3
- Urządzenia
  - przez sysfs
  - silniki
  - sensory
  - led
  - baterie
  - ... ;-)

Ralph Hempel

David Lechner

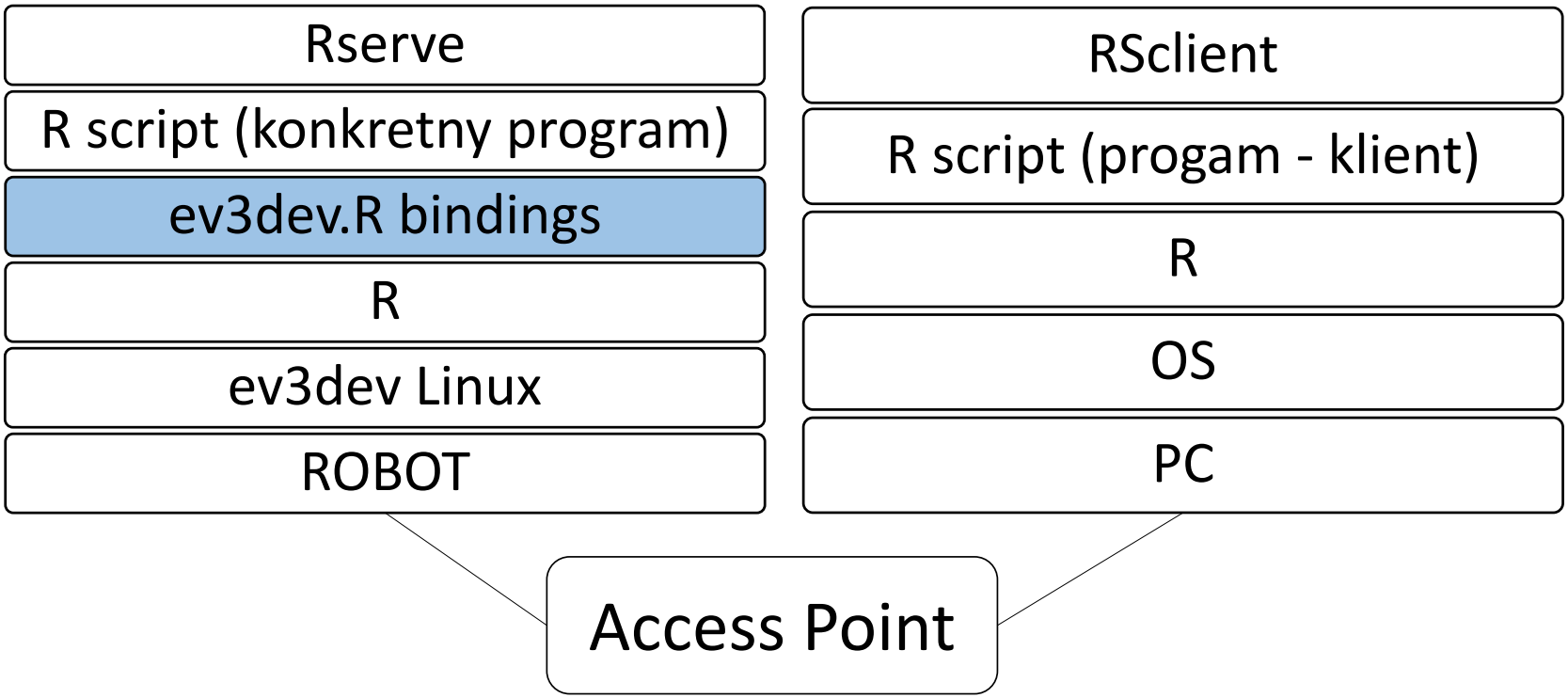
ev3dev sysfs - silnik



Przerywnik... ;-)

```
while(Value(infrared, 1L)!=-128L)
 {
   heading=Value(infrared)

   if(abs(heading)>forward_thr){
     target_speed=(heading-sign(heading)*forward_thr) / (2
     SetPulsesPerSecondSP(left, target_speed)
     SetPulsesPerSecondSP(right, -target_speed)
   }else{
     SetPulsesPerSecondSP(left, 0)
     SetPulsesPerSecondSP(right,0)
   }
   Sys.sleep(0.1)
 }
 StopEngines(left, right)
}
```

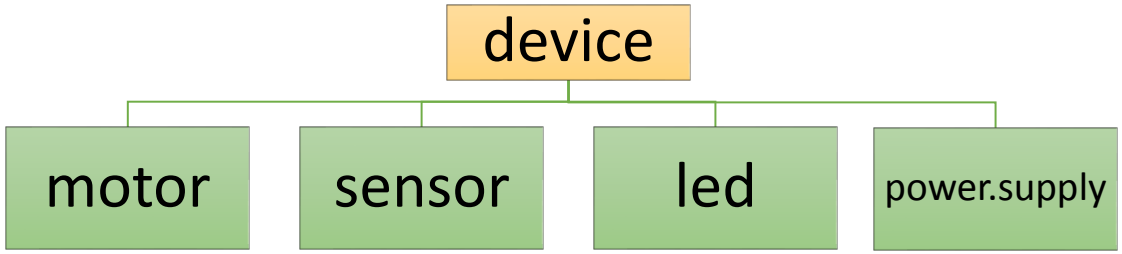
ev3dev.R bindings



Bindings w ev3dev

- Zunifikowane interfejsy
- Języki
  - C++
  - LUA
  - Node.js
  - Vala
  - R
- Inne (niezunifikowane)
  - Google Go
  - Python
  - C (+ Perl, Python, Ruby)
- `https://github.com/ev3dev/ev3dev-lang/blob/develop/wrapper-specification.md`

ev3dev.R bindings

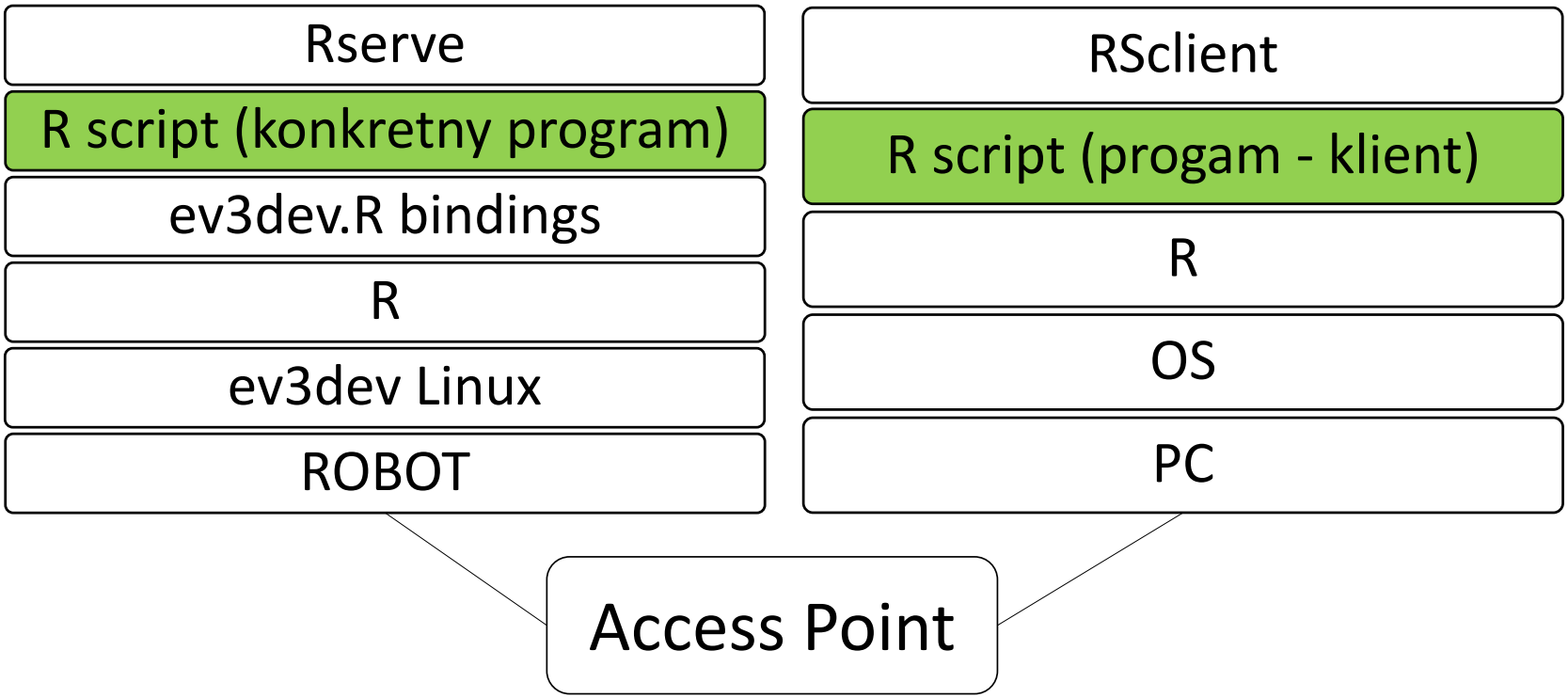


- Klasy S4
  - mutable
- Kod w R
- Przeznaczenie
  - kalibracja
  - prototypowanie
  - ...

ev3dev.R bindings - przykład

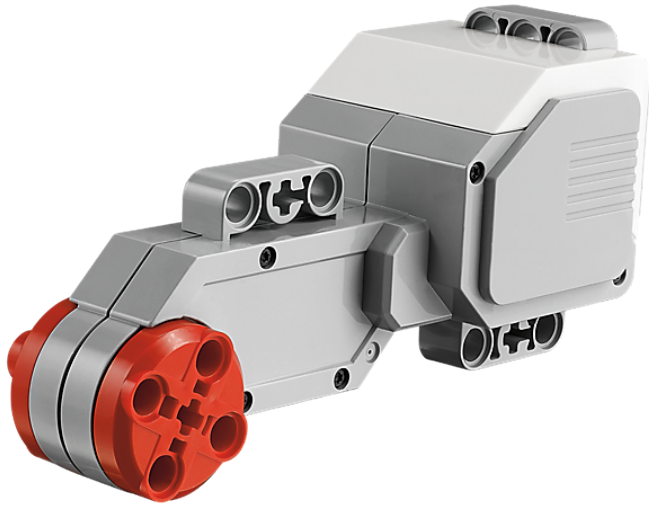
```
infrared=infrared.sensor(ports$INPUT_AUTO)

SetMode(infrared, 'IR-PROX')

Value(infrared)
```

ev3dev.R scripts



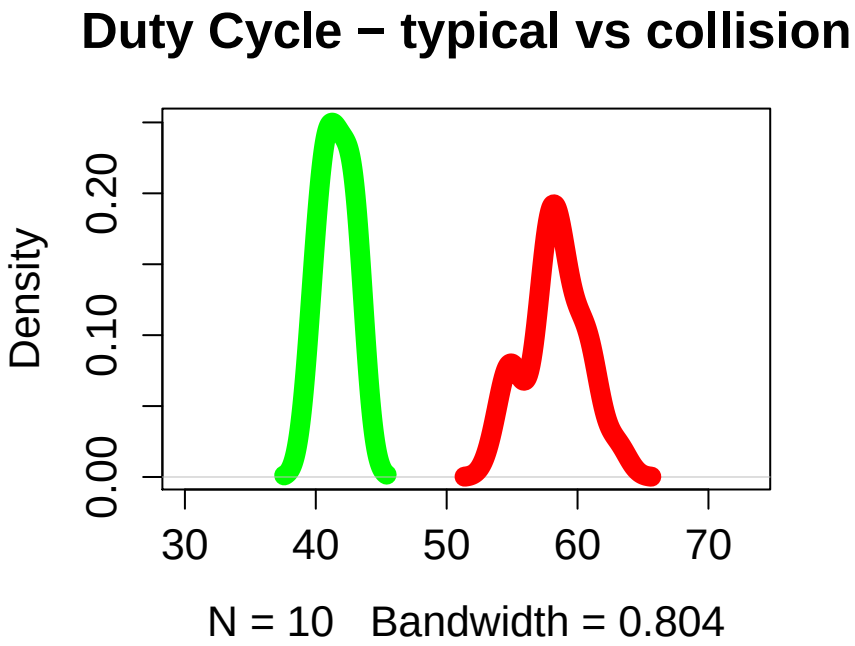
Dygresja - silniki

- Silniki
  - ...
  - serwomotory
    - enkodery - informacje zwrotna
    - pętla sprzężenia
    - dowolne sterowanie
  - ...
- Sterowanie
  - moment siły (prąd)
  - prędkość
  - pozycja
- Rampy

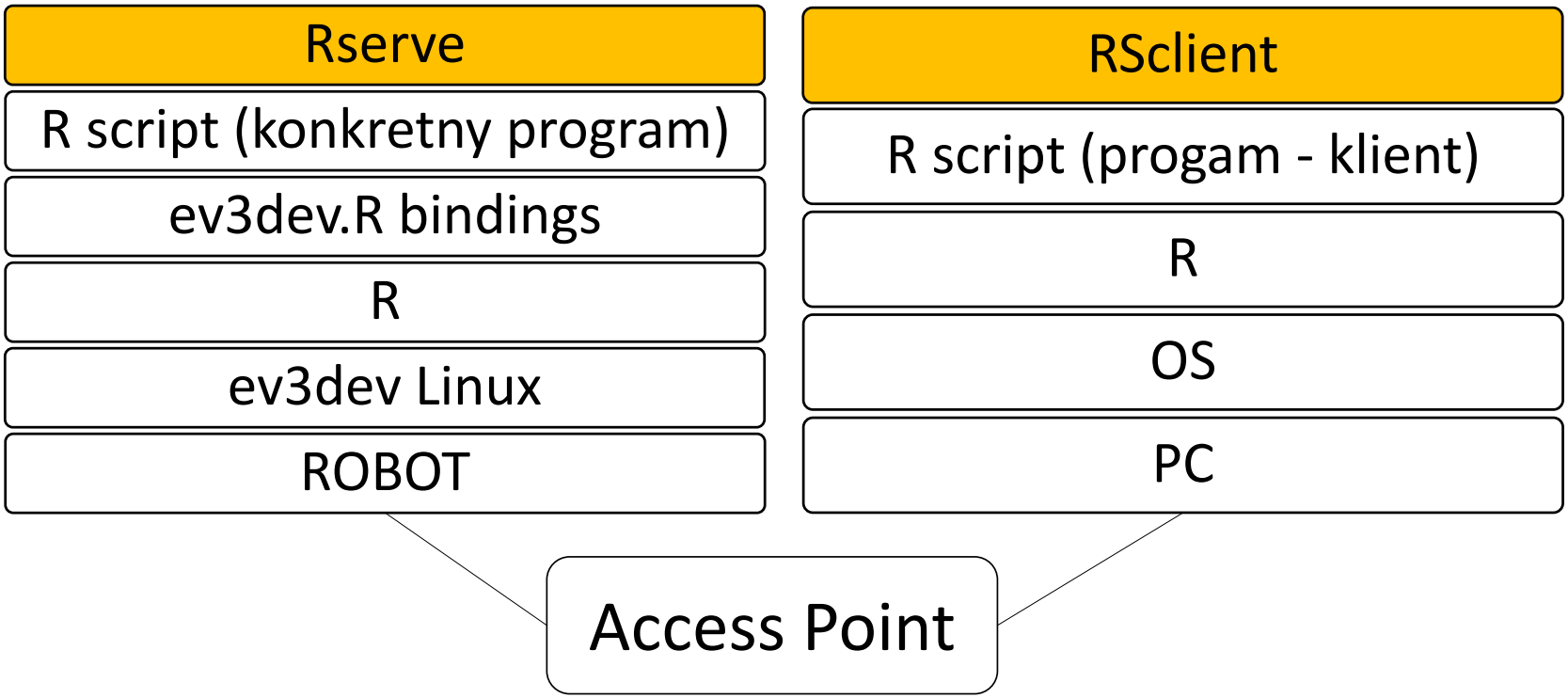


Przykład - przewidywanie kolizji

- W ev3dev DutyCycle
  - jak ciężko pracuje silnik
  - % U baterii -> silinka
  - zależy od naładowania baterii
- Kalibracja
  - normalna praca
  - kolizja



Rserve / RSclient



Rserve i RSclient

- Rserve
  - server TCP/IP
  - `http://rforge.net/Rserve/`
  - udostępnia R innym systemom
- RSclient
  - klient z R do Rserve
  - `http://www.rforge.net/RSclient/`
  - 2 interfejsy
    - np RSeval(...) - stary
    - np RS.eval(...) -nowy

Rserve i RSclient - przykład

```
library(RSclient)

con=RS.connect('192.168.1.10')

RS.eval(con, source("SER.R"))

# lazy evaluation
RS.eval(con, TestDutyCycle(20))

# mixing local/remote objects
cm=20
RS.eval( con, as.call( list(quote(TestDutyCycle), cm) ),
lazy=FALSE )

RS.close(con)
```

Rserve - przydatne funkcje

```
UploadFile=function(con, source_file,
 target_file=basename(source_file) )
{
  bytes=file.info(source_file)$size
  text=readChar(source_file, bytes)
  RS.eval(con, as.call(list(quote(writeChar), text,
   target_file, nchar(text) , NULL)), lazy=FALSE)
}

UploadFile( con, "SER.R" )
RS.eval( con, source("SER.R") )
```

Rserve - przydatne funkcje c.d.

```
DownloadFile=function(con, source_file,
 target_file=basename(source_file) )
{
  bytes=RS.eval(con, as.call(list(quote(file.info),
   source_file) ), lazy=FALSE)$size
  text=RS.eval(con, as.call(list(quote(readChar),
   source_file, bytes ) ), lazy=FALSE)
  writeChar(text, target_file, nchar(text) , NULL)
}

DownloadFile(con, "/var/log/Rserve.log")
print( readLines("Rserve.log") )
```

SLAM

- Simultaneous Localization And Mapping
- Potrzebujemy
  - mapy (żeby się zlokalizować)
  - lokalizacji (żeby zrobić mapę)
- Liczne
  - wersje (ziemia, woda, powietrze, ...)
  - rozwiązania (filtr Kalmana, filtry cząsteczkowe, ...)
  - wymagania (skala, dokładność, ...)
- Klasyczne: filtr Kalmana (EKF)

Dziękujemy za uwagę!

```
library(RSclient)

con=RS.connect(‘192.168.1.10‘)

RS.eval(con, SpeakPL(‘Dziękuję za uwagę!‘) )

RS.close(con)
```