Instytut Informatyki
Uniwersytet Wrocławski

Erlang - wprowadzenie

Marek Materzok

29 października 2007

Powody powstania (1979-82)

- Doświadczenia Bjarne'a Däckera z Lispem
  - Projektowanie układów elektronicznych
  - Kompilator dla urządzeń Ericcsona. *Interpreter w Lispie 7 do 10 razy krótszy niż w Pascalu, a wystarczająco wydajny*
- „Kryzys oprogramowania"
- EriPascal
  - Wzorowany na języku Chill, który był używany w telekomunikacji
  - Składnia Pascalowa, tylko przekazywanie komunikatów

Powody powstania (1984-86)

- Eksperymenty z językami - CSLab w Ericssonie
- Różne paradygmaty – programowanie imperatywne, funkcjonalne, obiektowe, programowanie w logice
- Konkluzja – *wielkoskalowa, drobnoziarnista równoległość* i *asynchroniczne przekazywanie komunikatów* okazały się bardzo pożądane

Pierwsze wersje Erlanga (1986-1990)

- Eksperymenty Joego Armstronga z Prologiem
  - Język zmieniał się w kierunku programowania funkcjonalnego
  - Powstały język został ochrzczony Erlang
- Pierwsze zastosowanie – programowanie centrali telefonicznych
- *XIII International Switching Symposium* ISS'90 – Erlang został pokazany światu
- Otwarcie źródeł – 1998

Wybrane zastosowania

- AXD 301 – przełączniki ATM Ericssona
  - Mieszanka języków – niski poziom w C, interfejs w Javie i JavaScripcie, sterowanie i zarządzanie w Erlangu
- ANx-DSL – urządzenia DSL dla dostawców Internetu
- GPRS
- Serwer XMPP ejabberd

Cechy

- Dynamicznie typowany
- Programowanie funkcjonalne, pattern matching
- Call-by-value, efekty uboczne, brak przypisania
- Wielkoskalowa wielozadaniowość, brak współdzielenia
- Łagodny czas rzeczywisty
- Rozproszenie, dynamiczna rekonfiguracja
- Odporność na awarie
- Hot-swapping kodu

Moduły i funkcje

```
-module(fib).
-export([fib/1, fibr/1]).

fibr(0) -> 0;
fibr(1) -> 1;
fibr(N) -> fibr(N-1) + fibr(N-2).

fib(0) -> 0;
fib(X) -> fib(0, 1, X).

fib(_, N, 1)    -> N;
fib(N1, N2, X) -> fib(N2, N1+N2, X-1).
```

Instytut Informatyki
Uniwersytet Wrocławski

Interpreter interaktywny

```
$ erl
Erlang (BEAM) emulator version 5.5.2

Eshell V5.5.2  (abort with ^G)
1> c(fib).
{ok,fib}
2> fib:fib(8).
21
3> fib:fibr(8).
21
4>
```

Krotki, atomy

```
-module(temp).
-export([convert/2]).

convert({U, V}, U) -> {U, V};
convert({cs, V}, fh) -> {fh, (9/5)*V+32};
convert({fh, V}, cs) -> {cs, (5/9)*(V-32)}.

Eshell V5.5.2  (abort with ^G)
1> c(temp).
{ok,temp}
1> temp:convert({cs,36.6},fh).
{fh,97.8800}
2>
```

Listy

```
-module(qsort).
-export([qsort/1]).

qsort([]) -> [];
qsort([H|T]) ->
    qsort([X||X <- T, X < H]) ++ [H] ++
    qsort([X||X <- T, X >= H]).

Eshell V5.5.2  (abort with ^G)
1> c(qsort).
{ok,qsort}
2> qsort:qsort([6,3,2,4]).
[2,3,4,6]
3>
```

Pattern matching

```
1> {A,B} = {foo,[1,2]}.
{foo,[1,2]}
2> {A,[C|D]} = {foo, [1,2]}.
{foo,[1,2]}
3> {A,B} = {bar, [1,2]}.

=ERROR REPORT==== 27-Oct-2007::13:48:49 ===
Error in process <0.30.0> with exit value:
{{badmatch,{bar,[1,2]}},[{erl_eval,expr,3}]}

** exited: {{badmatch,{bar,[1,2]}},[{erl_eval,expr,3}]} **
4>
```

Przekazywanie komunikatów

```
-module(primes).
-export([run/0, printer/0, filter/1, talker/2]).

run() ->
    Printer = spawn(primes, printer, []),
    Filter = spawn(primes, filter, [Printer]),
    spawn(primes, talker, [Filter, 2]).

printer() -> receive
        N -> io:format("~w~n", [N]), printer()
    end.

talker(Target, N) ->
    Target ! N,
    talker(Target, N+1).
```

Przekazywanie komunikatów, 2

```
filter(Target) -> receive
        N ->
            Target ! N,
            Filter = spawn(primes, filter, [Target]),
            filter(Filter, N)
    end.

filter(Target, N) -> receive
        K -> if
                K rem N /= 0 -> Target ! K;
                true -> nothing
            end,
            filter(Target, N)
    end.
```

Rozproszenie

```
-module(distr).
-export([start/0, proc/0]).

start() ->
    Pid = spawn(b@localhost,distr,proc,[]),
    Pid ! {self(), 'hello world'},
    receive X -> io:format("P1:~w~n", [X]) end.

proc() ->
    receive {Pid, X} ->
        io:format(user, "P2:~w~n", [X]), Pid ! X
    end.
```

Instytut Informatyki
Uniwersytet Wrocławski

Rozproszenie, 2

```
Eshell V5.5.2  (abort with ^G)
(a@localhost)1> c(distr).
{ok,distr}
(a@localhost)2> distr:start().
P1:'hello world'
ok
(a@localhost)3>

Eshell V5.5.2  (abort with ^G)
(b@localhost)1> P2:'hello world'
```

Obsługa wyjątków

```
Eshell V5.5.2  (abort with ^G)
1> catch(1).
1
2> catch(1/0).
{'EXIT',{badarith,[{erl_eval,eval_op,3},
                   {erl_eval,expr,5},
                   {shell,exprs,6},
                   {shell,eval_loop,3}]}}
3> catch(throw({wyjatek, bardzo_fajny})).
{wyjatek,bardzo_fajny}
4>
```

Monitorowanie procesów

```
-module(trap).
-export([start/0, fail/0]).

start() ->
    process_flag(trap_exit, true),
    spawn_link(trap, fail, []),
    receive
        {'EXIT', Pid, Reason} ->
            io:format("failure ~w: ~w~n", [Pid, Reason])
    end.

fail() -> throw(failure).
```

Monitorowanie procesów, 2

```
Eshell V5.5.2  (abort with ^G)
1> c(trap).
{ok,trap}
2> trap:start().

=ERROR REPORT==== 27-Oct-2007::15:03:03 ===
Error in process <0.37.0> with exit value:
{{nocatch,failure},[{trap,fail,0}]}

failure <0.37.0>: {{nocatch,failure},[{trap,fail,0}]}
ok
3>
```

Wymiana kodu w trakcie pracy

```
-module(codeswap).
-export([start/0, server/0, client/1]).
start() ->
    Pid = spawn(codeswap, server, []),
    spawn(codeswap, client, [Pid]).
server() -> receive
        M -> io:format("received ~w~n", [M]),
        server()
    end.
client(Pid) ->
    Pid ! 'hello',
    timer:sleep(1000),
    codeswap:client(Pid).
```

Wymiana kodu w trakcie pracy, 2

```
Eshell V5.5.2  (abort with ^G)
1> c(codeswap).
{ok,codeswap}
2> codeswap:start().
<0.38.0>received hello

received hello
received hello
3> c(codeswap).
{ok,codeswap}
received world
received world
```

Open Telecommunications Platform

- Framework do tworzenia dużych aplikacji
- Mnesia – rozproszona baza danych
- Implementacja SNMP
- Implementacja CORBA
- Kompilator ASN.1
- Obsługa XML

Słowo na koniec

- Zachęcam do samodzielnych eksperymentów z językiem
- `http://www.erlang.org/`