Razvoj programskog jezika Erlang

Seminarski rad u okviru kursa
Dizajn programskih jezika
Matematički fakultet

Igor Šćekić
igorscekic98@gmail.com

18. decembar 2019

Sažetak

U ovom radu prikazuju se osnove razvoja programskog jezika Erlang. Ukratko su opisani jezici koji su najviše uticali na njegov razvoj: Prolog, Lisp, Smalltalk i PLEX. Za svaki jezik date su osnovne informacije i način na koji je taj jezik uticao na osobine i razvoj jezika Erlang. Takođe, ukratko su opisani i jezici na koje je Erlang imao uticaj: Elixir i Scala. Prikazano je razvojno stablo programskog jezika Erlang.

Sadržaj

1 Uvod

Trenutno postoji više hiljada programskih jezika, i novi se stvaraju svake godine. Razlozi za nastanak novog jezika su, često, poboljšanje određenih koncepata, koji već postoje u nekim jezicima. Dakle, programski jezici nastaju pod uticajem svojih prethodnika, uzimajući njihove dobre osobine, i kombinujući ih. Na taj način se dobijaju jezici koji su veoma efikasni za rešavanje određene vrste problema, pa je stoga veoma važno odabrati pogodan jezik, u zavisnosti od toga šta treba isprogramirati.
Programski jezik **Erlang**, koji je nastao 1986. godine, je vrlo primenjiv kada postoji potreba da se implementiraju konkurentna i distribuirana rešenja. Koristi se na Eriksonovim telekomunikaconim sistemima i jako je skalabilan i siguran. I pored toga što je nastao u oblasti telekomunikacija, ima i širu upotrebu, pa je i uticao na nastanak modernih jezika, kao što su **Elixir** i **Scala**. Logo programskog jezika Erlang prikazan je na slici 1.

Slika 1: Logo programskog jezika Erlang

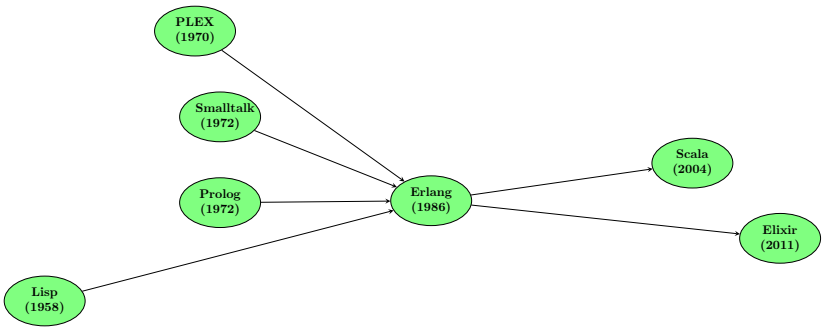
2 Osnovno o Erlangu

Erlang je funkcionalan, deklarativan i konkurentan programski jezik. Nastao je 1986. godine u laboratoriji Erikson CSLab (engl. *The Ericsson CSLab*), u okviru firme Erikson [9]. Zaposleni na ovom projektu želeli su da naprave novi programski jezik koji će biti najbolje rešenje za programiranje aplikacija za telekomunikacije. To su uradili dodavanjem konkurentnosti u programski jezik Prolog, zbog čega se Erlang jedno vreme smatrao dijalektom Prologa [1]. Nakon toga postao je potpuno novi programski jezik, veoma brzo se razvijao, nastajale su nove verzije, kao i standardna biblioteka **OTP** (engl. *The open Telecom Platform*).
Najzaslužniji za njegov nastanak i razvoj je informatičar Džo Armstrong (engl. *Joe Armstrong*).
Jedna od namena programskog jezika Erlang je pisanje što sigurnijih programa, koje je moguće popraviti bez isključivanja čitavog sistema [2].

3 Razvojno stablo

Na nastanak jezika Erlang je najviše uticao programski jezik **Lisp**. Pri njegovom kreiranju značajnu ulogu u smislu sintakse imao je programski jezik **Prolog**, a iz jezika **Smalltalk** i **PLEX** je preuzeo neke koncepte.

Uticao je na nastanak jezika **Elixir** i **Scala**. Razvojno stablo programskog jezika Erlang može se videti na slici 2.



Slika 2: Razvojno stablo jezika Erlang

3.1 Lisp

Programski jezik **Lisp** razvio je Džon Makarti (engl. *John McCarthy*) 1958. godine na Univerzitetu MIT [4]. U ovom jeziku veoma bitne strukture podataka su liste i drveta, zbog čega se jezik i zove Lisp (od *LISt Processing*). Čak mu je i sam izvorni kod sačinjen od listi, što je programerima otvorilo mogućnost za kreiranje makro sistema. Iako je Lisp opštenamenski programski jezik, obično se kaže da je jezik veštačke inteligencije, jer je to oblast u kojoj se najviše primenjuje.
Zasnovan je na Čerčovom formalizmu lambda računa, spada u grupu funkcionalnih programskih jezika i od njega su se dalje razvili svi savremeni funkcionalni programski jezici [4].
Lisp je uticao na Erlang funkcionalnom paradigmom [8].

3.2 Prolog

Jezik **Prolog** i prvi interpretator za njega razvijeni su na *Univerzitetu u Marseju* 1972. godine, kao plod saradnje istraživača Alain Colmerauera iz Marseja i Roberta Kowalskog iz Edinburga [5].
Najznačajniji je predstavnik logičke paradigme. Pogodan je za podršku relacionim bazama podataka, rešavanje problema matematičke logike, razumevanje prirodnih jezika i drugo. Vreme najveće popularnosti Prologa je prošlo, ali se on i dalje široko koristi, uglavnom za probleme iz oblasti veštačke inteligencije: od medicinskih sistema pa do istraživanja podataka [5]. Prolog je uticao na nastanak i razvoj mnogih programskih jezika. Konkretno, njegov uticaj na Erlang ogleda se u tome što sintaksa Erlanga veoma podseća na Prologovu: promenljive počinju velikim slovom, tačka se nalazi na kraju svake funkcionalne celine, oba jezika koriste poklapanje obrazaca (engl. *pattern matching*).

3.3 Smalltalk

Smalltalk je objektno-orijentisani, dinamički, reflektivni programski jezik. Nastao je 1972. godine, kao projekat kompanije Xerox PARC [7]. Ono što je karakteristično za ovaj programski jezik je to što je on čisto objektno-orijentisan. Čak i primitivni tipovi predstavljaju objekte klasa.

```
-module(hello).
-export([hello_world/0]).

hello_world() ->
  io:format("Hello, World!"),
  halt(0).
```

Listing 1: Erlang

```
hello :-
  format('Hello, World!'),
  halt.
```

Listing 2: Prolog

Slika 3: "Hello, world"program u jezicima Erlang i Prolog

Ovo omogućava dobru komunikaciju, putem poruka, sa instancama drugih klasa. U Smalltalk-u i sve klase su takođe objekti, tj. instance neke metaklase, a sve metaklase su instance klase *Metaclass*.
Smalltalk je reflektivan programski jezik jer ima sposobnost da ispituje sopstvenu strukturu. Jedan je od prvih programskih jezika sa grafičkim korisničkim okruženjem. Karakteriše ga jednostavna sintaksa[1] i nepostojanje *if*, *for* i *while* naredbi.
Erlang je preuzeo objektno-orijentisan koncept iz Smalltalk-a.

3.4 PLEX

PLEX (Programming Language for EXchanges) je programski jezik koji je, kao i Erlang, nastao u kompaniji Erikson. Dizajniran je 1970. godine specijalno za telefonske sisteme [6]. Zbog toga možemo reći da je jezik niskog nivoa (engl. *low-level programming language*), jer je usko vezan za hardver koji je korišćen u kompaniji Erikson. PLEX je koristio mešavinu zaštite hardvera (engl. *hardvare protection*) i kopiranja podataka, što je eliminisalo upotrebu pokazivača i mogućnosti za česte greške [1]. Upravo ovo je uticalo na postojanje sakupljača otpadataka (engl. *garbage collector*) u programskom jeziku Erlang, na koji je PLEX imao jak uticaj, jer je bio direktna inspiracija za njegov nastanak [1].

3.5 Scala

Scala je objektno-orijentisan programski jezik, nastao 2004. godine. Ima sintaksu koja podseća na sintaksu programskog jezika **C**. Scala ima jak sistem statičkih tipova i programi napisani u ovom programskom jeziku su veoma koncizni, tj. manjih dimenzija u odnosu na ostale programske jezike opšte namene. Sve biblioteke koje su podržane u Javi, mogu se koristiti i u Scali. Ime Scala je nastalo od *skalabilnosti* (engl. *scalability*) i *jezika* (engl. *language*), što znači da je dizajnirana da napreduje sa zahtevima svojih korisnika [12]. Izvorni kod Scale se kompajlira u Java bajt kod i izvršava se na Java virtuelnoj mašini, pa je pogodna i za razvoj android aplikacija [11]. Scala ima punu podršku za funkcionalno programiranje, pa se upravo tu ogleda uticaj Erlanga. Takođe, od Erlanga je nasledila i konkurentan koncept. Scala uživa veliku popularnost u modernom svetu IT kompanija.

[1] Ovo se može zaključiti i na osnovu njegovog imena: *Smalltalk* u prevodu sa engleskog jezika znači *ćaskanje*.

```erlang
-module(hello_module).
-export([some_fun/0,some_fun/1]).

% A "Hello world" function
some_fun() ->
    io:format('~s~n', ['Hello
     world!']).

% This one works only with lists
some_fun(List) when is_list(List)
    ->
    io:format('~s~n', List).

% Non-exported functions are
    private
priv() ->
    secret_info.
```

Listing 3: Erlang

```elixir
defmodule HelloModule do
    # A "Hello world" function
    def some_fun() do
        IO.puts "Hello World!"
    end

    # This one works only with
     lists
    def some_fun(list) when
     is_list(list) do
        IO.inspect list
    end

    # A private function
    defp priv do
        :secret_info
    end
end
```

Listing 4: Elixir

Slika 4: Poređenje sintakse jezika Erlang i Elixir

3.6 Elixir

Elixir je programski jezik koji pripada funkcionalnoj paradigmi. Dizajnirao ga je Hosé Valim (engl. *José Valim*), 2011. godine. Aplikacije napisane u Elixir-u su skalabilne i lake za održavanje. Zbog svoje funkcionalne prirode, izuzetno dobre podrške za rad u distribuiranim sistemima i tolerancije na greške koja je na jako visokom nivou, u Elixir-u je rađeno mnogo zanimljivih projekata iz oblasti robotike [3]. Elixir je napravljen da radi na virtuelnoj mašini jezika Erlang, a takođe je preuzeo i deo njegove standardne biblioteke. Pored toga što je Erlang uticao na njega, mnoge koncepte je preuzeo iz jezika Clojure, Haskell i Python, dok je programski jezik Ruby uticao na njegovu sintaksu. Elixir je danas veoma popularan programski jezik i mnoge kompanije ga svakodnevno koriste [10].

4 Zaključak

U ovom radu ukratko su predstavljene osnove razvoja programskog jezika Erlang. Prikazani su jezici koji su najviše uticali na njegov nastanak i razvoj, i prikazano je njegovo elementarno razvojno stablo. Date su informacije i o dva moderna programska jezika na čiji nastanak i razvoj je Erlang uticao. Za dodatne informacije i praćenje daljeg razvoja, kao početnu odrednicu najbolje je koristiti zvaničnu stranu programskog jezika Erlang [9], koja je i prilikom pisanja ovog rada najviše korišćena.

Literatura

[1] Joe Armstrong. A history of erlang. *HOPL*, 2007.

[2] Joe Armstrong. *Programming Erlang (2nd edition)*. Pragmatic Bookshelf, 2013.

[3] Milena Dukanac. Jezik Elixir sa primenom u sekvencioniranju genoma, 2019. Master rad.

[4] Predrag Janičić i Filip Marić. *Programiranje 2*. Matematički fakultet, Beograd, 2019.

[5] Predrag Janičić i Mladen Nikolić. *Veštačka inteligencija*. Matematički fakultet, Beograd, 2018.

[6] J.Erikson. *A Structural Operational Semantics for PLEX*. Technical report, Malardalen University, 2003.

[7] Alan C. Kay. *The Early History Of Smalltalk*. Apple Computer, 1993.

[8] Lisp. Zvanična stranica programskog jezika Lisp. https://lisp-lang.org/.

[9] Erlang OTP. Zvanična stranica programskog jezika Erlang. http://erlang.org.

[10] Charles Petit. A brief history of erlang and elixir. *Coder Stories*, 2019.

[11] Scala. Zvanična stranica programskog jezika Scala. https://www.scala-lang.org/.

[12] Joshua D. Suereth. *Scala in Depth*. Manning Publications, 2011.