Elixir Beginnings

THE JOY OF ELIXIR

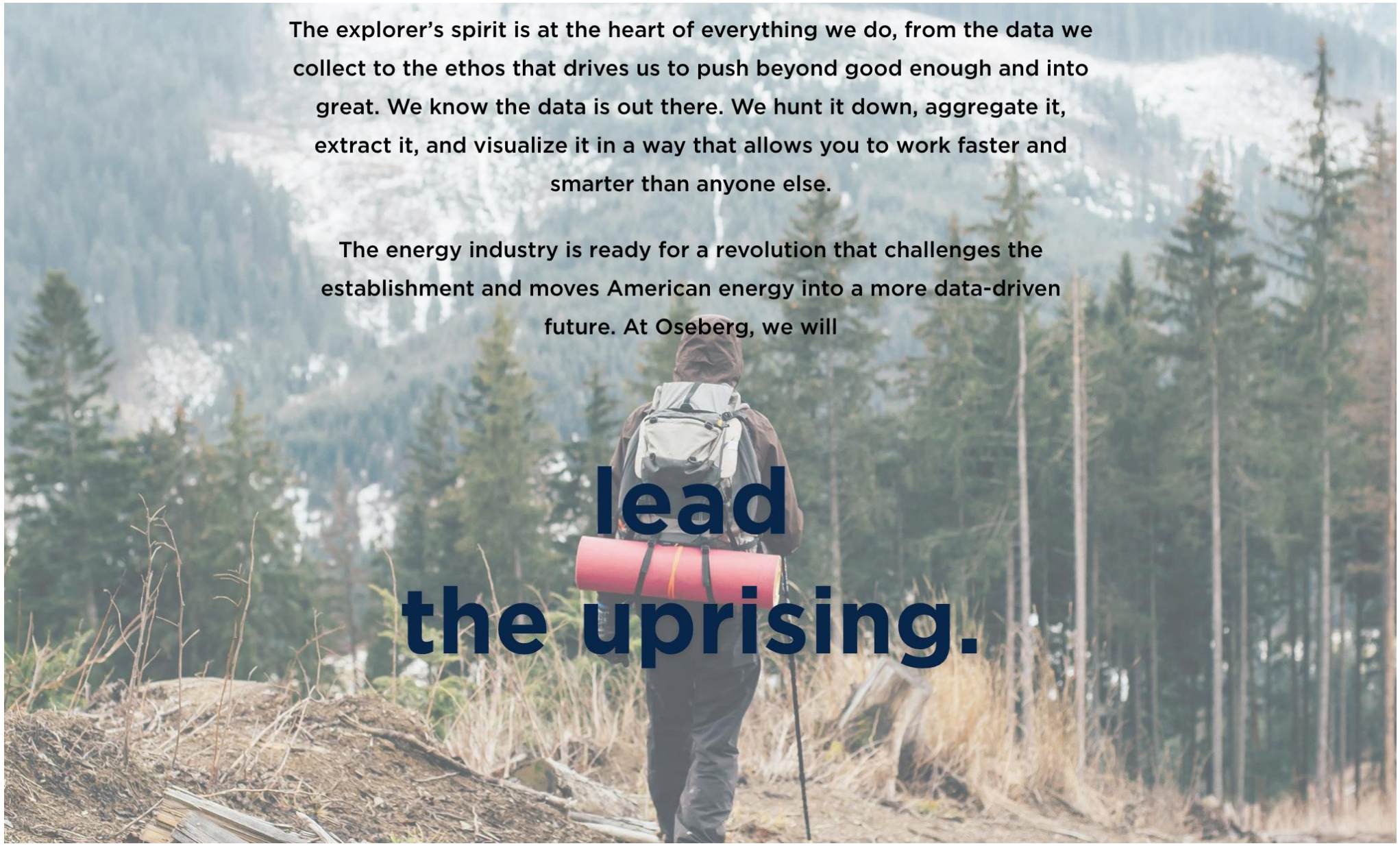
Thank you for the opportunity!

- Amazing community.
- Please donate.
- Thank you, sponsors!

oseberg

Sameeri Marryboyina

- SOFTWARE ENGINEER
- THE UNIVERSITY OF OKLAHOMA
- HYDERABAD, INDIA



The explorer's spirit is at the heart of everything we do, from the data we collect to the ethos that drives us to push beyond good enough and into great. We know the data is out there. We hunt it down, aggregate it, extract it, and visualize it in a way that allows you to work faster and smarter than anyone else.

The energy industry is ready for a revolution that challenges the establishment and moves American energy into a more data-driven future. At Oseberg, we will

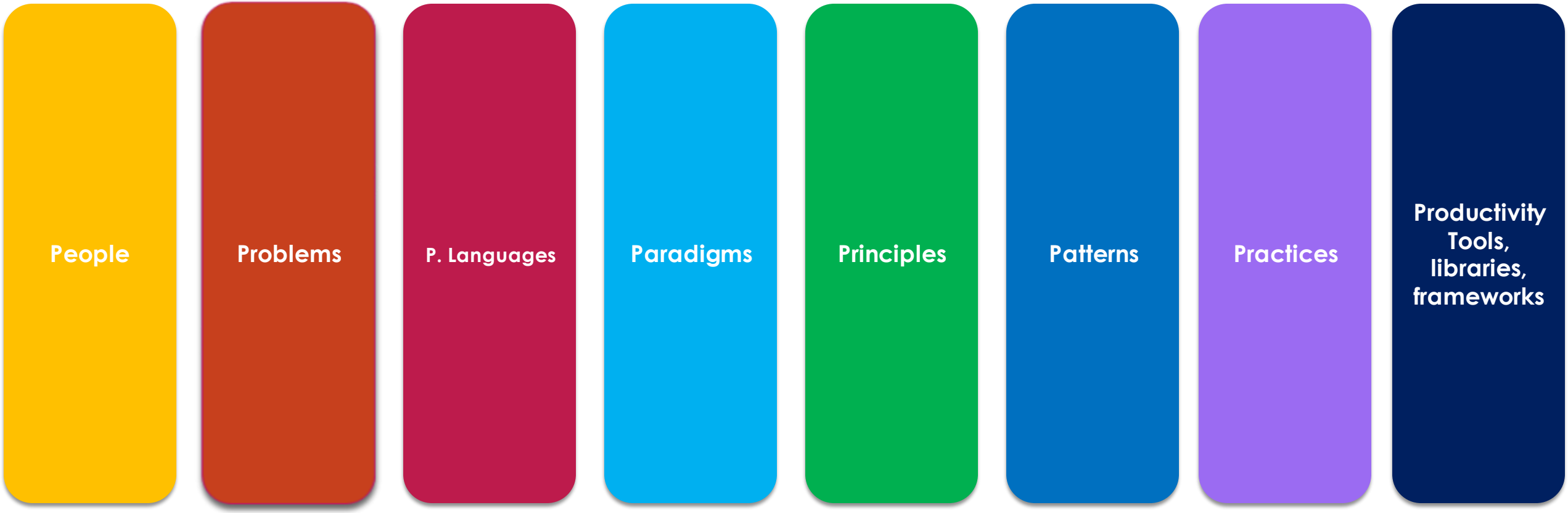
lead the uprising.

Join our Team!

TALK GOAL

- For the Love of Elixir
- To excite you and build an interest in Elixir, Erlang

Gedanken's Karte



My mind map to Software Engineering

Language Characteristics

Creators, Paradigms, Typing Discipline, Influenced by, Influenced

Lisp

Paradigm	Multi-paradigm: functional, procedural, reflective, meta
Designed by	John McCarthy
Developer	Steve Russell, Timothy P. Hart, and Mike Levin
First appeared	1958; 60 years ago
Typing discipline	dynamic, strong

Dialects

Arc · AutoLISP · Clojure · Common Lisp · Emacs Lisp · EuLisp · Franz Lisp · Hy · Interlisp · ISLISP · LeLisp · LFE · Maclisp · MDL · newLISP · NIL · Picolisp · Portable Standard Lisp · Racket · RPL · Scheme · SKILL · Spice Lisp · T · Zetalisp

Influenced by

IPL

Influenced

CLIPS · CLU · COWSEL · Dylan · Elixir · Falcon · Forth · Haskell · Io · Ioke · JavaScript · Julia[1] · Logo · Lua · ML · Nim · Nu · OPS5 · Perl · POP-2/11 · Python · R · Rebol · Ruby · Scala · Swift · Smalltalk · Tcl · Wolfram Language[2]

C

THE C PROGRAMMING LANGUAGE

The C Programming Language[1] (often referred to as "K&R"), the seminal book on C

Paradigm	Imperative (procedural), structured
Designed by	Dennis Ritchie
Developer	Dennis Ritchie & Bell Labs (creators); ANSI X3J11 (ANSI C); ISO/IEC JTC1/SC22/WG14 (ISO C)
First appeared	1972; 46 years ago[2]
Stable release	C11 / December 2011; 6 years ago
Typing discipline	Static, weak, manifest, nominal
OS	Cross-platform
Filename extensions	`.c`, `.h`

Major implementations

K&R, GCC, Clang, Intel C, Microsoft Visual C++, Pelles C, Watcom C

Dialects

Cyclone, Unified Parallel C, Split-C, Cilk, C*

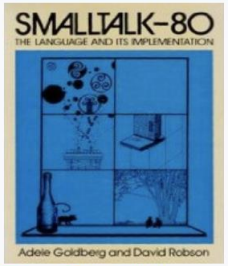
Influenced by

B (BCPL, CPL), ALGOL 68,[3] Assembly, PL/I, FORTRAN

Influenced

Numerous: AMPL, AWK, csh, C++, C--, C#, Objective-C, D, Go, Java, JavaScript, Julia, Limbo, LPC, Perl, PHP, Pike, Processing, Python, Ring[4], Rust, Seed7, Vala, Verilog (HDL)[5]

Smalltalk



Paradigm	Object-oriented
Designed by	Alan Kay, Dan Ingalls, Adele Goldberg
Developer	Alan Kay, Dan Ingalls, Adele Goldberg, Ted Kaehler, Diana Merry, Scott Wallace, Peter Deutsch and Xerox PARC
First appeared	1972; 46 years ago (development began in 1969)
Stable release	Smalltalk-80 version 2 / 1980; 38 years ago
Typing discipline	Strong, dynamic
OS	Cross-platform (multi-platform

Major implementations

Amber, Dolphin Smalltalk, GemStone/S, GNU Smalltalk, Pharo, Smalltalk/X, Squeak, VA Smalltalk, VisualWorks

Influenced by

Lisp,[1] Simula,[1] Euler,[1] IMP,[1] Planner,[1] Logo,[2] Sketchpad,[1] ARPAnet,[1] Burroughs B5000,[1] cell (biology)[1]

Influenced

AppleScript, Common Lisp Object System, Dart, Dylan, Erlang, Etoys, Falcon, Go, Groovy, Io, Ioke, Java, Lasso, Lisaac, Logtalk, Newspeak, NewtonScript, Object REXX, Objective-C, PHP 5, Perl 6, Python, Ruby, Scala, Scratch, Self

C++

Paradigm	Multi-paradigm: procedural, functional, object-oriented, generic[1]
Designed by	Bjarne Stroustrup
First appeared	1985; 33 years ago
Stable release	ISO/IEC 14882:2017 / 1 December 2017; 4 months ago
Typing discipline	Static, nominative, partially inferred
Implementation language	C++ or C
Filename extensions	.C .cc .cpp .cxx .c++ .h .hh .hpp .hxx .h++
Website	isocpp.org

Major implementations

LLVM Clang, GCC, Microsoft Visual C++, Embarcadero C++Builder, Intel C++ Compiler, IBM XL C++, EDG

Influenced by

Ada, ALGOL 68, C, CLU, ML, Simula

Influenced

Ada 95, C#,[2] C99, Chapel,[3] D, Java,[4] Lua, Perl, PHP, Python,[5] Rust, Nim[*citation needed*]

Erlang

Paradigm	multi-paradigm: concurrent, functional
Designed by	Joe Armstrong, Robert Virding, and Mike Williams
Developer	Ericsson
First appeared	1986; 32 years ago
Stable release	20.3[1] / 14 March 2018; 27 days ago
Typing discipline	dynamic, strong
License	Apache License 2.0 (since OTP 18.0) Erlang Public License 1.1 (earlier releases)
Filename extensions	.erl .hrl
Website	www.erlang.org

Major implementations

Erlang

Influenced by

Prolog, Smalltalk, PLEX,[2] LISP

Influenced

F#, Clojure[*citation needed*], Rust, Scala, Opa, Reia, Elixir, Dart, Akka, Oz

Haskell

Paradigm	functional, lazy/non-strict, modular
Designed by	Lennart Augustsson, Dave Barton, Brian Boutel, Warren Burton, Joseph Fasel, Kevin Hammond, Ralf Hinze, Paul Hudak, John Hughes, Thomas Johnsson, Mark Jones, Simon Peyton Jones, John Launchbury, Erik Meijer, John Peterson, Alastair Reid, Colin Runciman, Philip Wadler
First appeared	1990; 28 years ago[1]
Stable release	Haskell 2010[2] / July 2010; 7 years ago
Preview release	Haskell 2020 announced[3]
Typing discipline	static, strong, inferred
OS	Cross-platform
Filename extensions	`.hs`, `.lhs`
Website	www.haskell.org

Major implementations

GHC, Hugs, NHC, JHC, Yhc, UHC

Dialects

Helium, Gofer

Influenced by

Clean,[4] FP,[4] Gofer,[4] Hope and Hope+,[4] Id,[4] ISWIM,[4] KRC,[4] Lisp,[4] Miranda,[4] ML and Standard ML,[4] Orwell, SASL,[4] Scheme,[4] SISAL[4]

Influenced

Agda,[5] Bluespec,[6] C++11/Concepts,[7] C#/LINQ,[8][9][10][11] CAL,[*citation needed*] Cayenne,[8] Clean,[8] Clojure,[12] CoffeeScript,[13] Curry,[8] Elm, Epigram,[*citation needed*] Escher,[14] F#,[15] Frege,[16] Hack,[17] Idris,[18] Isabelle,[8] Java/Generics,[8] LiveScript,[19] Mercury,[8] Omega,[*citation needed*] Perl 6,[20] PureScript,[21] Python,[8][22] Rust,[23] Scala,[8][24] Swift,[25] Timber,[26] Visual Basic 9.0[8][9]

Python

Paradigm	Object-oriented, imperative, functional, procedural, reflective
Designed by	Guido van Rossum
Developer	Python Software Foundation
First appeared	1990[1]
Stable release	3.6.5 / 28 March 2018; 22 days ago[2] 2.7.14 / 16 September 2017; 6 months ago[3]
Preview release	3.7.0b3[4] / 29 March 2018; 21 days ago
Typing discipline	Duck, dynamic, strong
License	Python Software Foundation License
Filename extensions	.py, .pyc, .pyd, .pyo (prior to 3.5),[5] .pyw, .pyz (since 3.5)[6]
Website	python.org

Major implementations

CPython, IronPython, Jython, MicroPython, Numba, PyPy, Stackless Python

Dialects

Cython, RPython

Influenced by

ABC,[7] ALGOL 68,[8] C,[9] C++,[10] CLU,[11] Dylan,[12] Haskell,[13] Icon,[14] Java,[15] Lisp,[16] Modula-3,[10] Perl

Influenced

Boo, Cobra, Coconut,[17] CoffeeScript,[18] D, F#, Falcon, Genie,[19] Go, Groovy, JavaScript,[20][21] Julia,[22] Nim, Ring,[23] Ruby,[24] Swift[25]

JS

Paradigm	Multi-paradigm: object-oriented (prototype-based), imperative, functional, event-driven[1]
Designed by	Brendan Eich
Developer	Netscape Communications Corporation, Mozilla Foundation, Ecma International
First appeared	December 4, 1995; 22 years ago[2]
Stable release	ECMAScript 2017[3] / June 2017; 10 months ago
Typing discipline	Dynamic, duck
Filename extensions	`.js`
Website	Mozilla

Major implementations

V8, JavaScriptCore, SpiderMonkey, Chakra

Influenced by

Java, Lua, Scheme, Perl, Self, C, Python, AWK, HyperTalk

Influenced

ActionScript, AtScript, CoffeeScript, Dart, JScript .NET, LiveScript, Objective-J, Opa, Perl 6, QML, TypeScript

Language Characteristics

Creators, Paradigms, Typing Discipline, Influenced by, Influenced

Paradigm	Multi-paradigm: Object-oriented, imperative, functional, reflective
Designed by	Yukihiro Matsumoto
Developer	Yukihiro Matsumoto, *et al.*
First appeared	1995; 23 years ago
Stable release	2.5.1 (March 28, 2018; 23 days ago[1]) [±]
Typing discipline	Duck, dynamic, strong
Scope	Lexical, sometimes dynamic
Implementation language	C
OS	Cross-platform
License	Ruby, GPLv2 or 2-clause BSD license[2][3][4]
Filename extensions	.rb
Website	www.ruby-lang.org

Major implementations

Ruby MRI, YARV, Rubinius, MagLev, JRuby, MacRuby, RubyMotion, Mruby

Influenced by

Ada,[5] C++,[5] CLU,[6] Dylan,[6] Eiffel,[5] Lisp,[6] Lua, Perl,[6] Python,[6] Smalltalk[6]

Influenced

Clojure, CoffeeScript, Crystal, D, Elixir, Falcon, Groovy, Ioke,[7] Julia,[8] Mirah, Nu,[9] Reia, Ring,[10] Rust, Swift[11]

C#

Paradigm	Structured, imperative, object-oriented, event-driven, task-driven, functional, generic, reflective, concurrent
Family	C
Designed by	Microsoft
Developer	Microsoft
First appeared	2000; 18 years ago[1]
Stable release	7.2[2] / November 15, 2017; 4 months ago
Preview release	8.0[3]
Typing discipline	static, dynamic,[4] strong, safe, nominative, partially inferred
Platform	Common Language Infrastructure
License	CLR: MIT/X11[5] Mono compiler: dual GPLv3 and MIT/X11 Libraries: LGPLv2 DotGNU: dual GPL and LGPLv2
Filename extensions	.cs
Website	docs.microsoft.com/dotnet/csharp/language-reference/

Major implementations

Visual C#, .NET Framework, Mono, DotGNU

Dialects

Cω, Spec#, Polyphonic C#, Enhanced C#

Influenced by

C++,[6] Eiffel, Java,[6] Modula-3, Object Pascal,[7] ML, VB, Icon, Haskell, Rust, J#, Cω, F#,[8] J++

Influenced

Chapel,[8] Crystal,[9] D, J#, Dart,[10] F#, Hack, Java,[11][12] Kotlin, Monkey, Nemerle, Oxygene, Ring[13], Rust, Swift,[14] Vala

Scala

Paradigm	Multi-paradigm: functional, object-oriented, imperative, concurrent
Designed by	Martin Odersky
Developer	Programming Methods Laboratory of École Polytechnique Fédérale de Lausanne
First appeared	20 January 2004; 14 years ago
Stable release	2.12.5 / 15 March 2018; 37 days ago[1]
Typing discipline	Static, strong, inferred, structural
Implementation language	Scala
Platform	JVM, JavaScript,[2] LLVM[3] (experimental)
License	BSD 3-clause[4]
Filename extensions	.scala, .sc
Website	www.scala-lang.org

Influenced by

Eiffel, Erlang, Haskell,[5] Java,[6] Lisp,[7] Pizza,[8] Standard ML,[6] OCaml,[6] Scheme,[6] Smalltalk, Oz

Influenced

Ceylon, Fantom, F#, Kotlin, Lasso, Red

Paradigm	functional
Designed by	Rich Hickey
First appeared	2007; 11 years ago
Stable release	1.9[1] / December 8, 2017; 4 months ago
Typing discipline	dynamic, strong
Platform	JVM, CLR, JavaScript
License	Eclipse Public License
Filename extensions	.clj, .cljs, .cljc, .edn
Website	clojure.org

Influenced by

C++,[2] C#, Common Lisp, Erlang, Haskell, Mathematica,[3] ML, Prolog, Scheme, Java, Racket,[4] Ruby[5]

Influenced

Elixir, Hy, Pixie, Rhine

Go's mascot is a gopher, designed by Renée French.[1]

Paradigm	Multi-paradigm: procedural, functional,concurrent
Designed by	Robert Griesemer Rob Pike Ken Thompson
Developer	Google
First appeared	November 10, 2009; 8 years ago
Stable release	1.10.1 / March 28, 2018; 22 days ago[2]
Typing discipline	Strong, static, inferred, structural[3][4]
Implementation language	Go, assembly language (gc); C++ (gccgo)
OS	DragonFly BSD, FreeBSD, Linux, macOS, NetBSD, OpenBSD,[5] Plan 9,[6] Solaris, Windows
License	BSD-style[7] + patent grant[8]
Filename extensions	.go
Website	golang.org

Major implementations

gc, gccgo

Influenced by

Alef, APL,[9] BCPL,[9] C, CSP, Limbo, Modula, Newsqueak, Oberon, occam, Pascal,[10] Python,[11] Smalltalk[12]

Influenced

Crystal

Paradigm	Multi-paradigm: compiled, concurrent, functional, imperative, structured, generic
Designed by	Originally Graydon Hoare, then Rust project developers
Developer	Rust project developers
First appeared	2010; 8 years ago
Stable release	1.25.0[1] / March 29, 2018; 16 days ago
Typing discipline	Static, strong, inferred, nominal, linear
Implementation language	Rust
OS	Linux, macOS, Windows, FreeBSD, OpenBSD,[2] Redox (operating system) Android, iOS (partial)[3]
License	MIT License or Apache License 2.0[4]
Filename extensions	.rs, .rlib
Website	www.rust-lang.org

Influenced by

Alef,[5] C#,[5] C++,[5] Cyclone,[5][6] Erlang,[5] Haskell,[5] Haxe,[5] Hermes,[5] Limbo,[5] Newsqueak,[5] NIL,[5] OCaml,[5] Ruby,[5] Scheme,[5] Standard ML,[5] Swift[5][7]

Influenced

Crystal, Elm,[8] Idris[9]

elixir

Paradigm	multi-paradigm: functional, concurrent, distributed, process-oriented
First appeared	2011; 7 years ago
Stable release	1.6.4 / 16 March 2018; 25 days ago[1]
Typing discipline	dynamic, strong
Platform	Erlang
License	Apache License 2.0[2]
Filename extensions	.ex, .exs
Website	elixir-lang.org

Influenced by

Erlang, Ruby, Clojure

Influenced

LFE

What it means to be a Language…

What problem are we solving?

- A language is a tool.
- Problems and Contexts
- We want to use the right tool for a given problem
- Communication

Nature

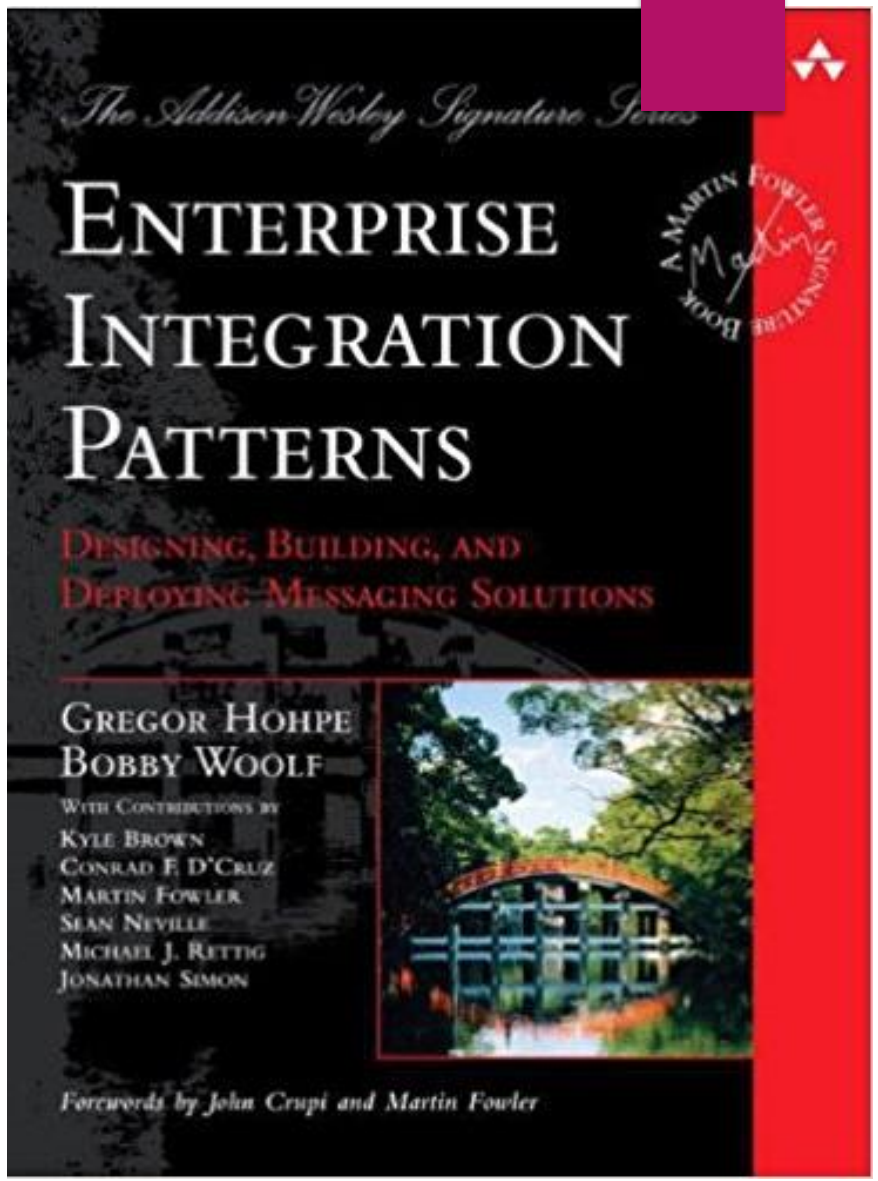
- Phillosophy
- Alphabet
- Constructs
- Syntax
- Semantics
- Type System (Static vs Dynamic)
- Translation Model (Compiled vs Interpreted)
- Paradigms
- Execution model
- Runtime Mechanics
- Specification
- Implementation

Key Takeaways

- Languages influence one another
- Each paradigm presents a way to think about the context and the problem
- Some problems are solved by many languages. This is good. We have choices.
- Modern languages manage memory for us.
- People are opinionated.

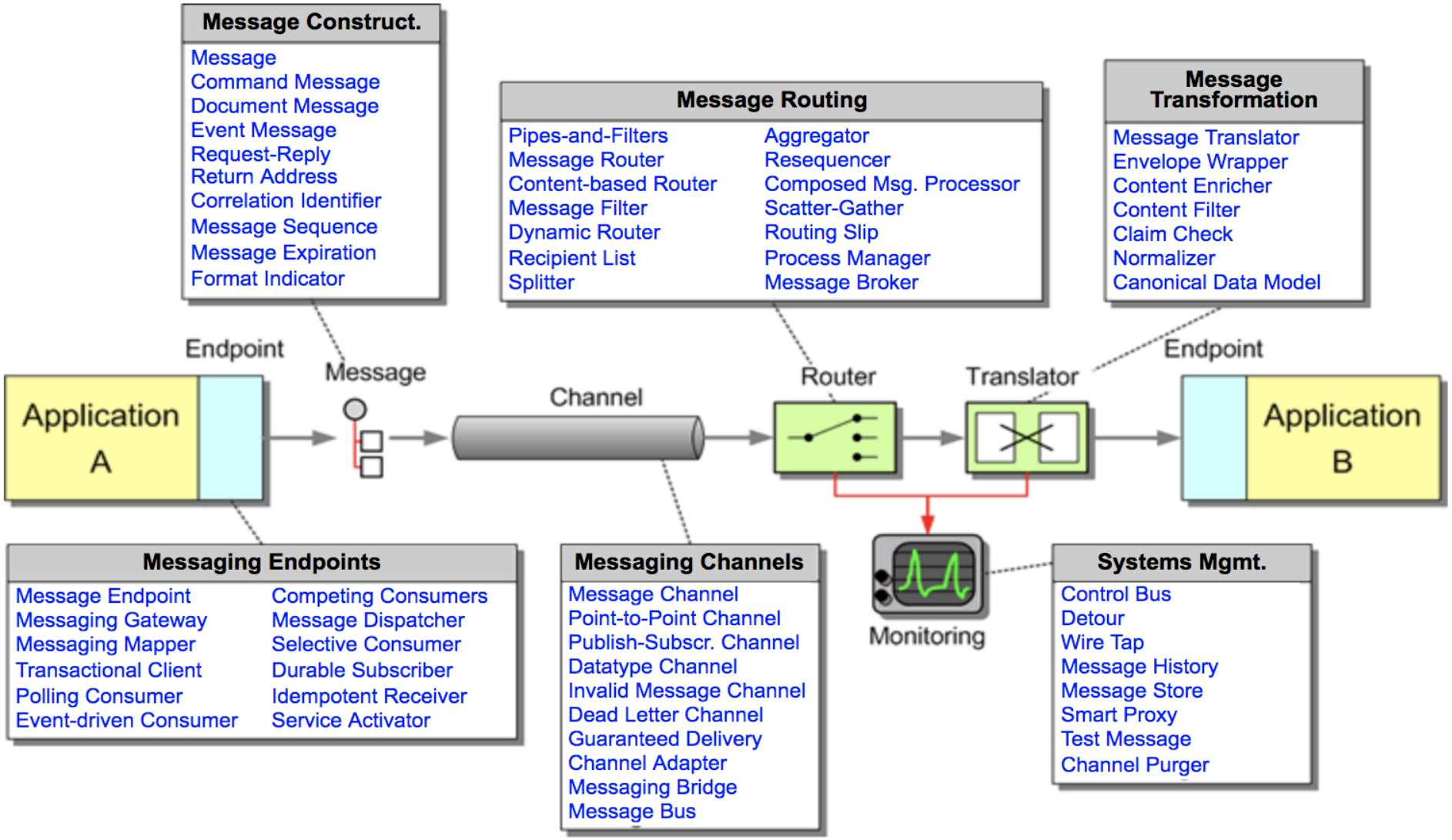
Enterprise Integration Patterns

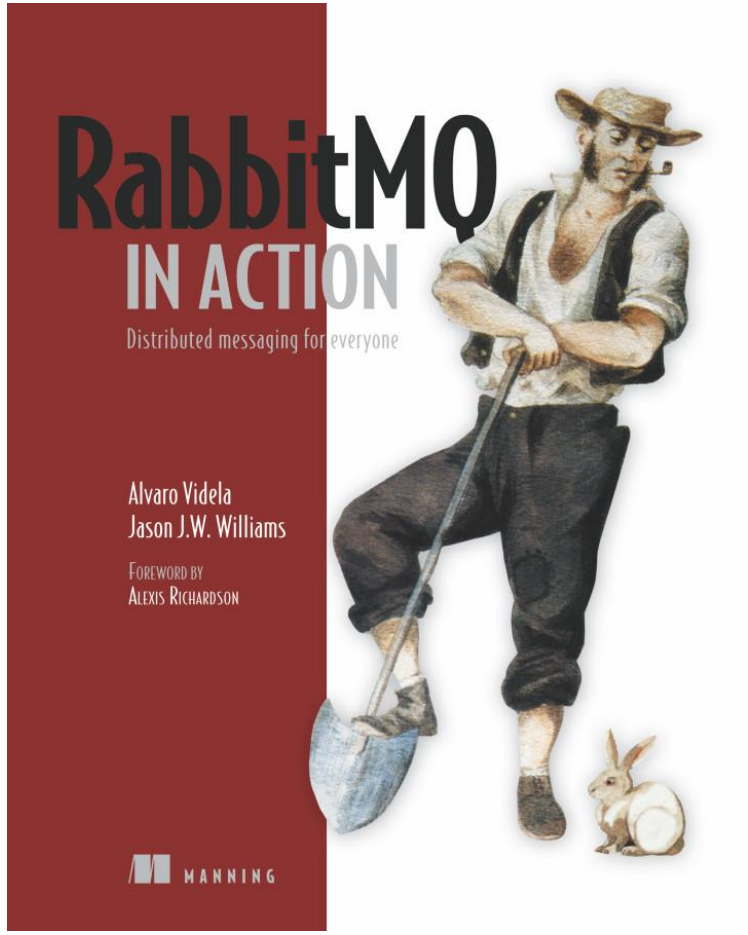
- How do we connect Systems across multiple machines?
- We need special components to make systems talk to each other.
- What are the patterns/vocabulary?
- Message Oriented Architecture/ Message based Systems
- The core construct is a "Message".



http://www.enterpriseintegrationpatterns.com/patterns/messaging/

We have documented 65 messaging patterns, organized as follows:





RabbitMQ

MESSAGE BROKER

JP MORGAN CHASE

AMQP

ERLANG

ERLANG USERS

- RabbitMQ
- CouchDB
- Watsapp
- AdRoll
- Basho Riak
- Klarna
- Amazon SimpleDB

INSPIRED BY ERLANG, ACTOR MODEL

- Akka
- Service Fabric

The Watsapp Story

CNN tech BUSINESS CULTURE GADGETS FUTURE STARTUPS

Facebook buys WhatsApp for $19 billion

by Adrian Covert @CNNTech

February 19, 2014: 6:54 PM ET

WhatsApp with Facebook's $19B offer?

In a play to dominate messaging on phones and the Web,

The WhatsApp Architecture Facebook Bought For $19 Billion

WEDNESDAY, FEBRUARY 26, 2014 AT 8:56AM

Rick Reed in an upcoming talk in March titled That's 'Billion' with a 'B': Scaling to the next level at WhatsApp reveals some eye popping WhatsApp stats:

> *What has hundreds of nodes, thousands of cores, hundreds of terabytes of RAM, and hopes to serve the billions of smartphones that will soon be a reality around the globe? The Erlang/FreeBSD-based server infrastructure at WhatsApp. We've faced many challenges in meeting the ever-growing demand for our messaging services, but as we continue to push the envelope on size (>8000 cores) and speed (>70M Erlang messages per second) of our serving system.*

But since we don't have that talk yet, let's take a look at a talk Rick Reed gave two years ago on WhatsApp: Scaling to Millions of Simultaneous Connections.

Facebook Closes $19 Billion W

WhatsApp founders Brian Acton and Jan Koum at their headquarters in Mountain View. (Photo by Robert Gallagher for Forbes)

Parmy Olson, FORBES STAFF

AI, bots and emerging tech in Europe. FULL BIO

Facebook says it has wrapped up its landmark $19 billion acquisition of WhatsApp, a deal that was hashed out in Mark Zuckerberg's house over the course of a few days in February and sealed over a bottle of Jonnie Walker scotch.

CADE METZ BUSINESS 09.15.15 07:00 AM

WHY WHATSAPP ONLY NEEDS 50 ENGINEERS FOR ITS 900M USERS

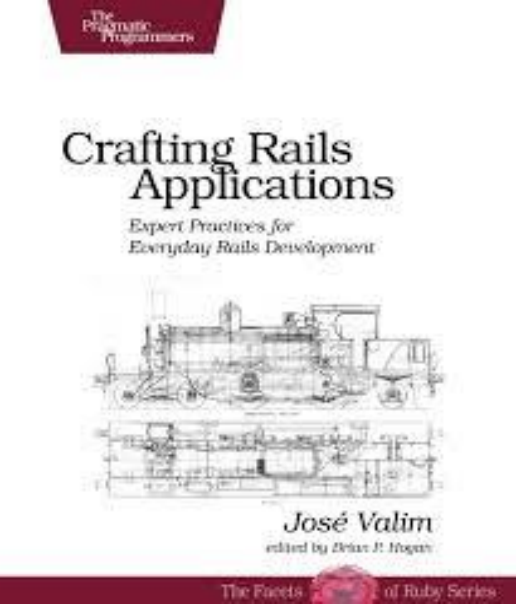
EARLIER THIS MONTH, in a post to his Facebook page, WhatsApp CEO Jan Koum announced that his company's instant messaging service is now used by more than 900 million people. And then Facebook CEO Mark Zuckerberg promptly responded with two posts of his own. One said "congrats," and the other included a cheeky photo Zuckerberg had taken of Koum as the WhatsApp CEO keyed his 900-million-user post into a smartphone. "Here's an action shot of you writing this update," Zuckerberg wrote.

WhatsApp is owned by Facebook, after Zuckerberg and

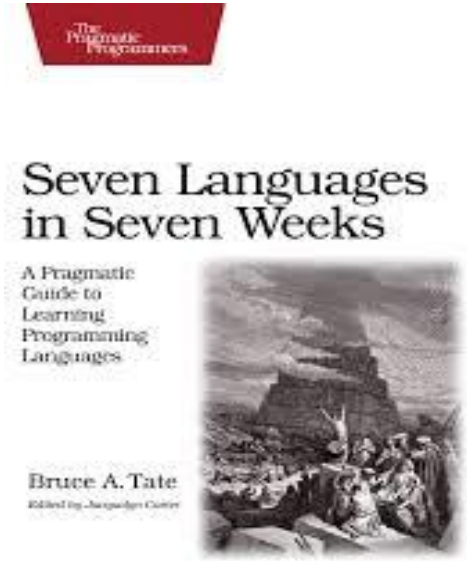
ELIXIR HISTORY

JOSE VALIM

- Brazilian Engineer
- Ruby on Rails Core Maintainer



3. Researches alternate technologies, languages for high-scalable systems



4. Discovers Erlang via Seven Languages book

2. Investigates scalability with concurrency. Thinks threads are not cool!

ERLANG

6. Misses tooling based productivity

5. Learns and works with Erlang. Loves it.

elixir

1. Jose' Valim
Ruby on Rails
Core maintainer

7. Implements Elixir to have all of Erlang's advantages, but with added productivity.

ERLANG HISTORY

A history of Erlang – JoeArmstrong

http://webcem01.cem.itesm.mx:8005/erlang/cd/downloads/hopl_erlang.pdf

The CS Lab

http://webcem01.cem.itesm.mx:8005/erlang/cd/downloads/hopl_erlang.pdf

- Ericsson – Stockholm, Sweden
- Bjarne Däcker brings along Mike Williams
- 1983 – The CS Lab is established
- State of the Art – AXE telephone exchange written in a proprietary language PLEX
- Hire Joe Armstrong, Rob Virding
- Bjarne Däcker to the team : "Solve Ericsson's Software problem"
- Investigate various languages.
- Go to conferences – Ask questions about Failure
- Look at University papers.
- Have access to a VMS, telephone switch, and UNIX.
- Joe starts with Smalltalk.
- Develops his own hand-written graphic notation for basic telephony.
- Roger Skagervall looks at Joe's algebra and introduces him to Prolog.
- Joe starts working in Prolog.
- He develops a meta-interpreter in Prolog.
- The meta-interpreter starts to grow in size.
- Joe wants to add concurrency.

The team

- What sarts as an experiment in “adding concurrency to Prolog” becomes more of a language in its own right.
- This language acquired a name “Erlang,”
- Deliberately encourage the ambiguity in the name – ERricsson's LANGuage vs Agner Krarup Erlang
- The meta-interpreter gets a name – JAM (Joe's Abstract Machine)
- Develop a philosophy - Concurrency Oriented Programming
- Robert Virding wants to play with Joe's code.
- Both review, argue and improve the code.
- Rob works in Parlog, in parallel. And he builds his own system.
- Introduce Erlang to users.
- Collaborate with users. Train them. Remove language features if not necessary.
- Work with The University of Uppasla. (Fondly referred to Uppasla boys)
- Go to conferences, give talks.
- Write a book.

BEAM

- A new system called AXE-N gets started.
- C++ is used to program the AXE-N.
- The project fails. Miserably.
- A new Erlang team gets started on it.
- The team change priorities from promotion to writing code.
- Prolog based implementation is slow.
- A C based implementation is created. TEAM(Turbo Erlang Abstract Machine)
- Somehow, the name changes from TEAM to BEAM(Bogdan's Erlang Abstract Machine)
- The team realize that they need a suite of principles, best-practices, conventions, tools, abstractions.
- A new Product team called OTP is formed, since resources with the Erlang team are limited.

Bluetail, AB

- Political reasons drive Erlang to be banned for internal usage.
- Later, they open source Erlang with some influence from Jane.
- All the original people of the team leave and form Bluetail AB with Jane as CEO.
- They use the experience, and Erlang to build internet based products.

1	Handling a very large number of concurrent activities
2	Actions to be performed at a certain point of time or within a certain time
3	Systems distributed over several computers
4	Interaction with hardware
5	Very large software systems
6	Complex functionality such as feature interaction
7	Continuous operation over several years
8	Software maintenance (reconfiguration, etc.) without stopping the system
9	Stringent quality and reliability requirements
10	Fault tolerance both to hardware failures and software errors

Table 1. Requirements of a programming language for telecommunication switching systems (from [12]).

Figure 6. Early internal marketing – the relationship between Erlang and PLEX.

erlang vsn 1.05

h	help
⊛**reset**	reset all queues
reset_erlang	kill all erlang definitions
load(F)	load erlang file <F>.erlang
load	load the same file as before
load(?)	what is the current load file
what_erlang	list all loaded erlang files
go	reduce the main queue to zero
send(A,B,C)	perform a send to the main queue
send(A,B)	perform a send to the main queue
cq	see queue - print main queue
wait_queue(N)	print wait_queue(N)
cf	see frozen - print all frozen states
eqns	see all equations
eqn(N)	see equation(N)
start(Mod,Goal)	starts Goal in Mod
top	top loop run system
q	quit top loop
open_dots(Node)	opens Node
talk(N)	N=1 verbose, =0 silent
peep(M)	set peeping point on M
no_peep(M)	unset peeping point on M
vsn(X)	erlang vsn number is X

Figure 2. The Erlang 1.05 manual.

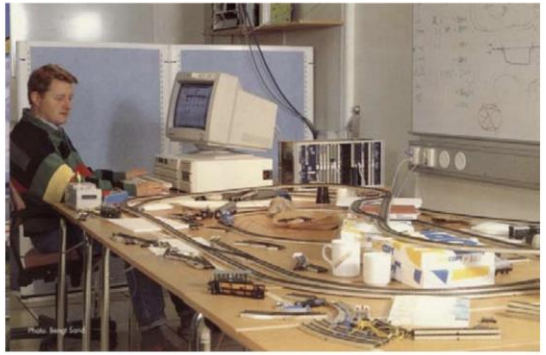
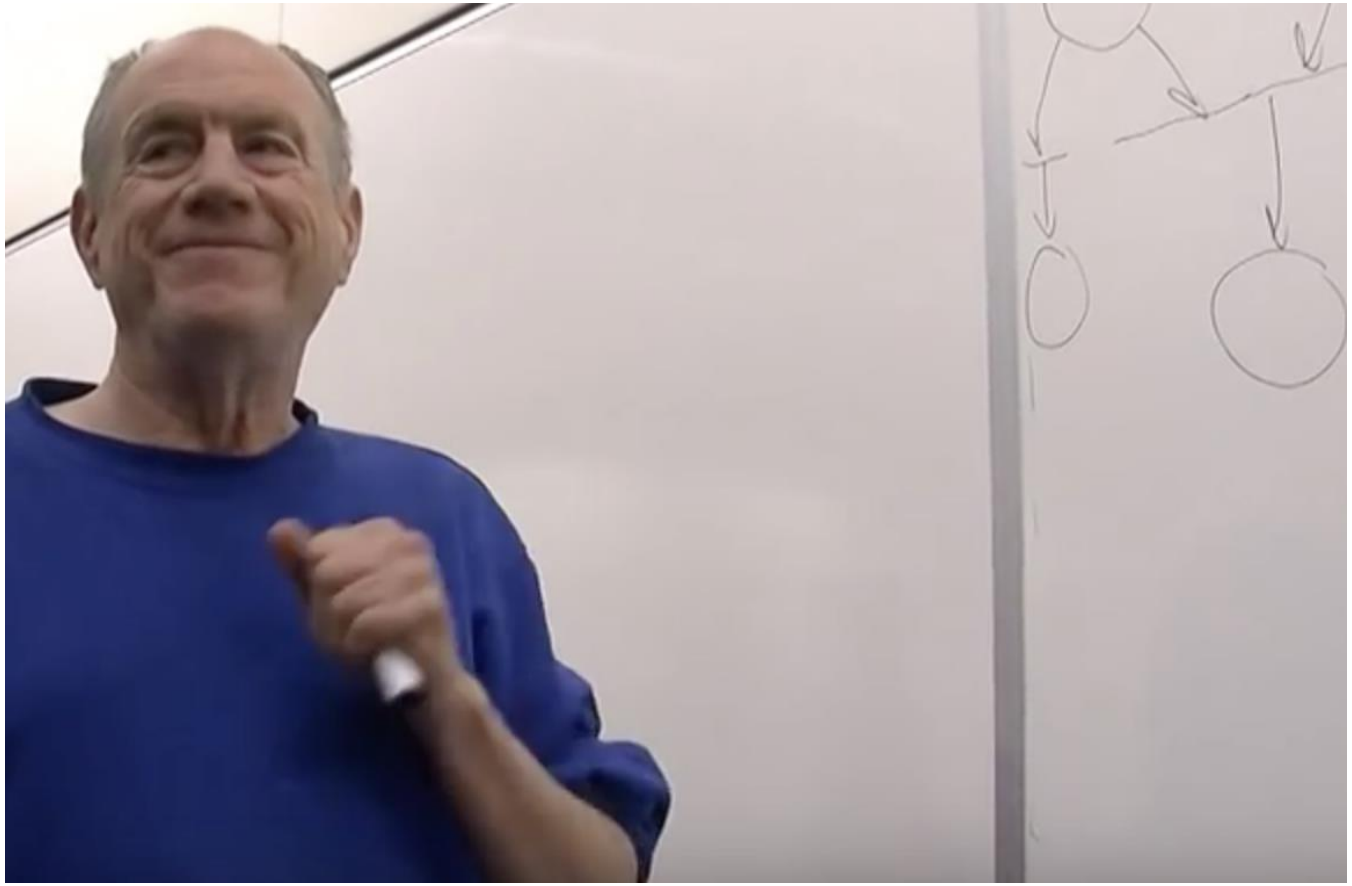


Figure 8. Robert Virding hard at work in the lab (1993).

THE ACTOR MODEL

"Carl Hewitt explaining the Actor model" - https://www.youtube.com/watch?v=7erJ1DV_Tlo&t=761s

- **The actor model adopts the philosophy that everything is an actor.**
- This is similar to the everything is an object philosophy used by some object-oriented programming languages.
- **An actor is a computational entity that, in response to a message it receives, can concurrently:**
- send a finite number of messages to other actors;
- create a finite number of new actors;
- designate the behavior to be used for the next message it receives.
- There is no assumed sequence to the above actions and they could be carried out in parallel.
- Decoupling the sender from communications sent was a fundamental advance of the Actor model enabling asynchronous communication and control structures as patterns of passing messages.[8]
- Recipients of messages are identified by address, sometimes called "mailing address". Thus an actor can only communicate with actors whose addresses it has. It can obtain those from a message it receives, or if the address is for an actor it has itself created.
- The actor model is characterized by inherent concurrency of computation within and among actors, dynamic creation of actors, inclusion of actor addresses in messages, and interaction only through direct asynchronous message passing with no restriction on message arrival order.



Demos

HTTPS://GITHUB.COM/SAMEERI/ELIXIR-BEGINNINGS

Thank you!


Appreciate me
promise to always show you gratitude, humility, and understanding because we care about you!
Support me
We promise to be your home away from home so you always feel comfortable and loved.
Delight me
We promise to go above and beyond to make you feel special and cared for.
Leave us a NOTE
love to hear from you!
Tell us...
*how was your drink?
*how are we doing?
*how was your experience?
...we want to know!
Our store manager
Amanda
would LOVE to hear your
-questions
-comments
-concerns
e-mail her at US1383433@starbucks
Amanda
Martin
Eric
Vivi
Morgan
Victoria
Sean
Alyssa
Zoey
Javier
Lauren
LaShawn
Eileen
Jazmin
Stefanie
Gabe
Jenny
Alex
Jose
Nicole
Skyler