

LAMBDA DAYS 2020

BEHIND F# EDITOR TOOLING



IONIDE
SATURN
SAFE STACK
F# TOOLING
OPEN SOURCE SOFTWARE
PHOTOGRAPHY

KRZYSZTOF CIESLAK
@K_CIESLAK



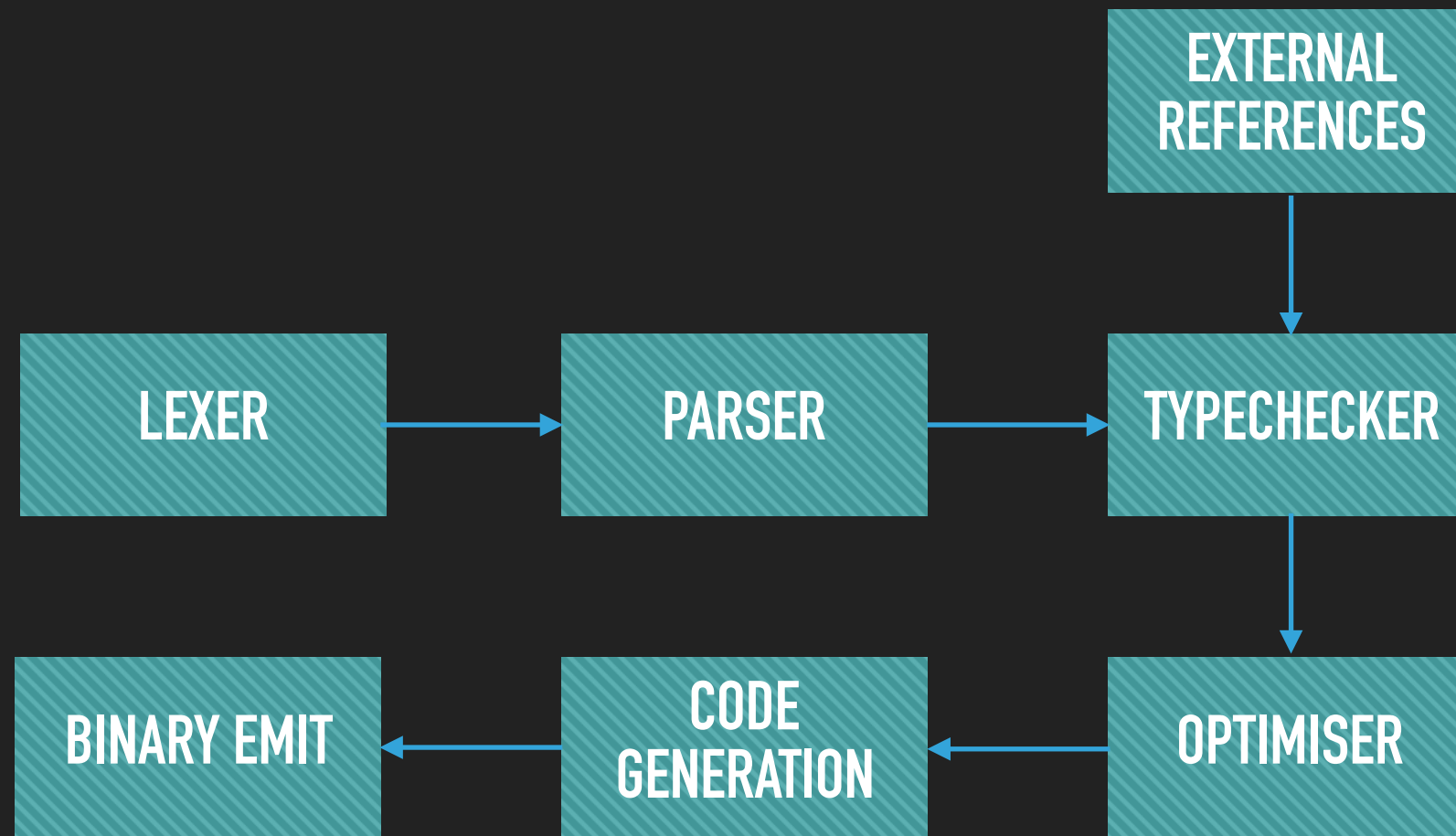
- ▶ COMPILER DESIGN
 - ▶ COMPILER AS A SERVICE
 - ▶ LANGUAGE SERVER PROTOCOL
 - ▶ USER EXPERIENCE
 - ▶ F# TOOLING ARCHITECTURE
-

**WHAT I'M GOING
TO TALK ABOUT?**

GLOSSARY

- ▶ F# - Functional first language running on .Net
- ▶ VS - Visual Studio, IDE from Microsoft for Windows
- ▶ VSCode - Visual Studio Code, cross-platform editor/IDE
- ▶ FCS - F# Compiler Service
- ▶ FSAC - FsAutoComplete, language server for F#
- ▶ AST - Abstract Syntax Tree
- ▶ TAST - Typed Abstract Syntax Tree

COMPILERS 101





RICH

UNINTRUSIVE

CORRECT

CONTEXT-AWARE

RESPONSIVE

PERFORMANT

**MODERN EDITOR
TOOLING**



BLACK BOX, BATCH MODE
APPROACH IS NOT GOOD
ENOUGH BASE FOR
MODERN EDITOR TOOLING

**COMPILER 101
IS NOT ENOUGH**



COMPILER NEEDS TO BE:

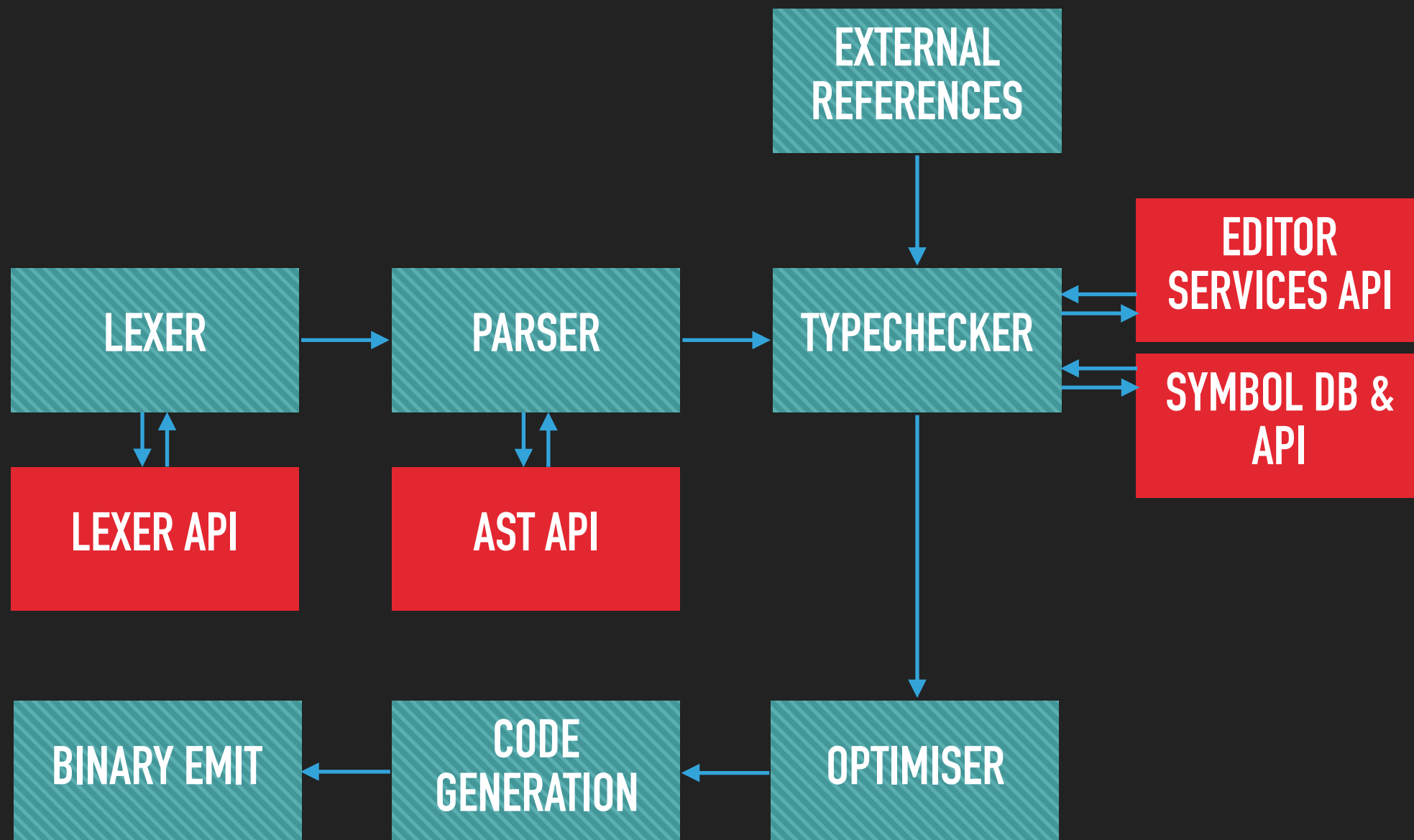
- * SERVER
- * API
- * DATABASE

**THIS HEAVILY IMPACTS
COMPILER DESIGN**

VERSION OF F# COMPILER WITH
ADDITIONAL APIS

FSHARP.COMPILER.SERVICE

COMPILER SERVICE DESIGN





LOT OF PATTERN MATCHING
USED TO IMPLEMENT FEATURES
BASED ON CODE STRUCTURE

AST API



BUILT-IN, LOW LEVEL EDITOR
FUNCTIONALITIES

BASE FOR MANY EDITOR FEATURES

**EDITOR SERVICE
API**



RICH INFORMATION ABOUT ENTITIES PRESENT IN
YOUR CODE

USED TO ENRICH INFORMATION PRESENTED TO
THE USER

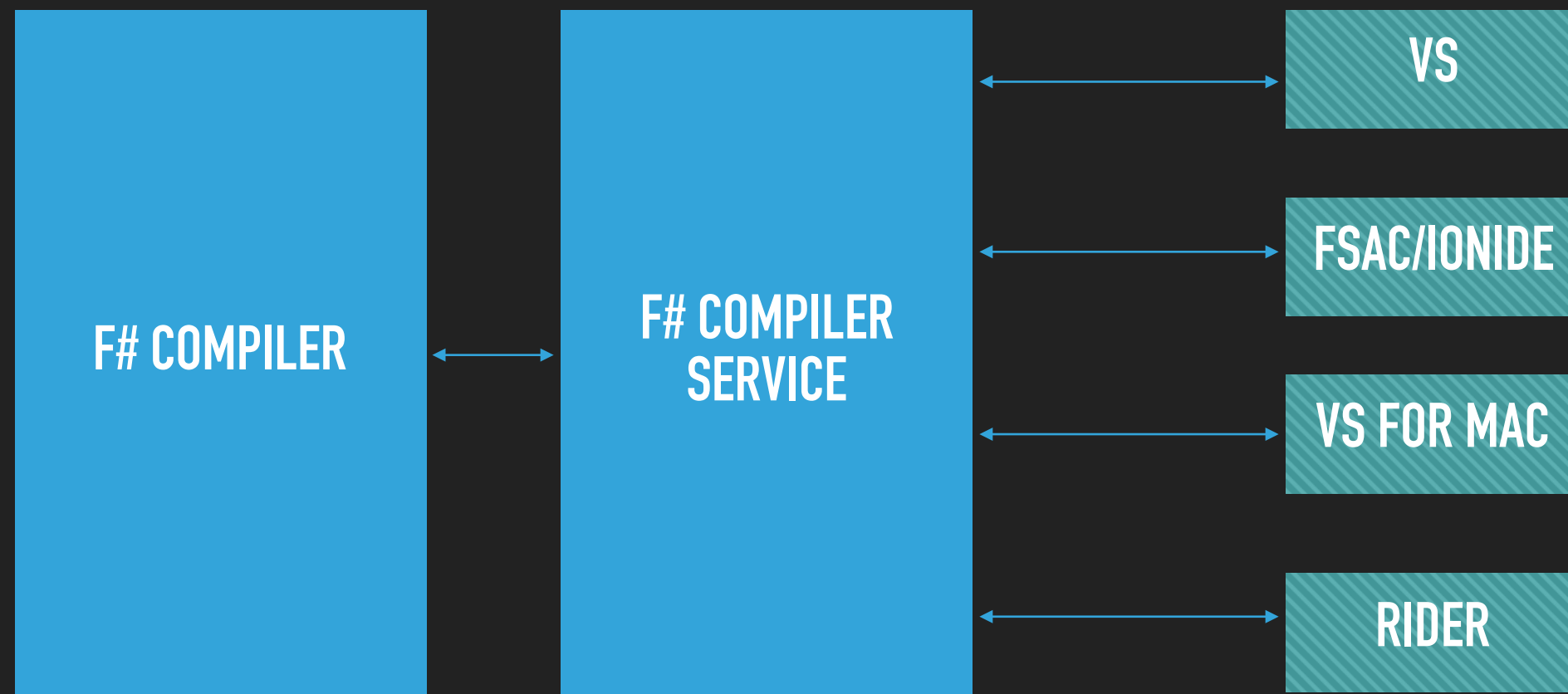
ACCESS TO TYPE INFORMATION

SYMBOL API

**FSHARP.COMPILER.SERVICE
IS NORMAL .NET LIBRARY**

UBIQUITOUS LANGUAGE TOOLING

UBIQUITOUS LANGUAGE TOOLING





COMMUNITY DRIVEN OSS
PROJECT

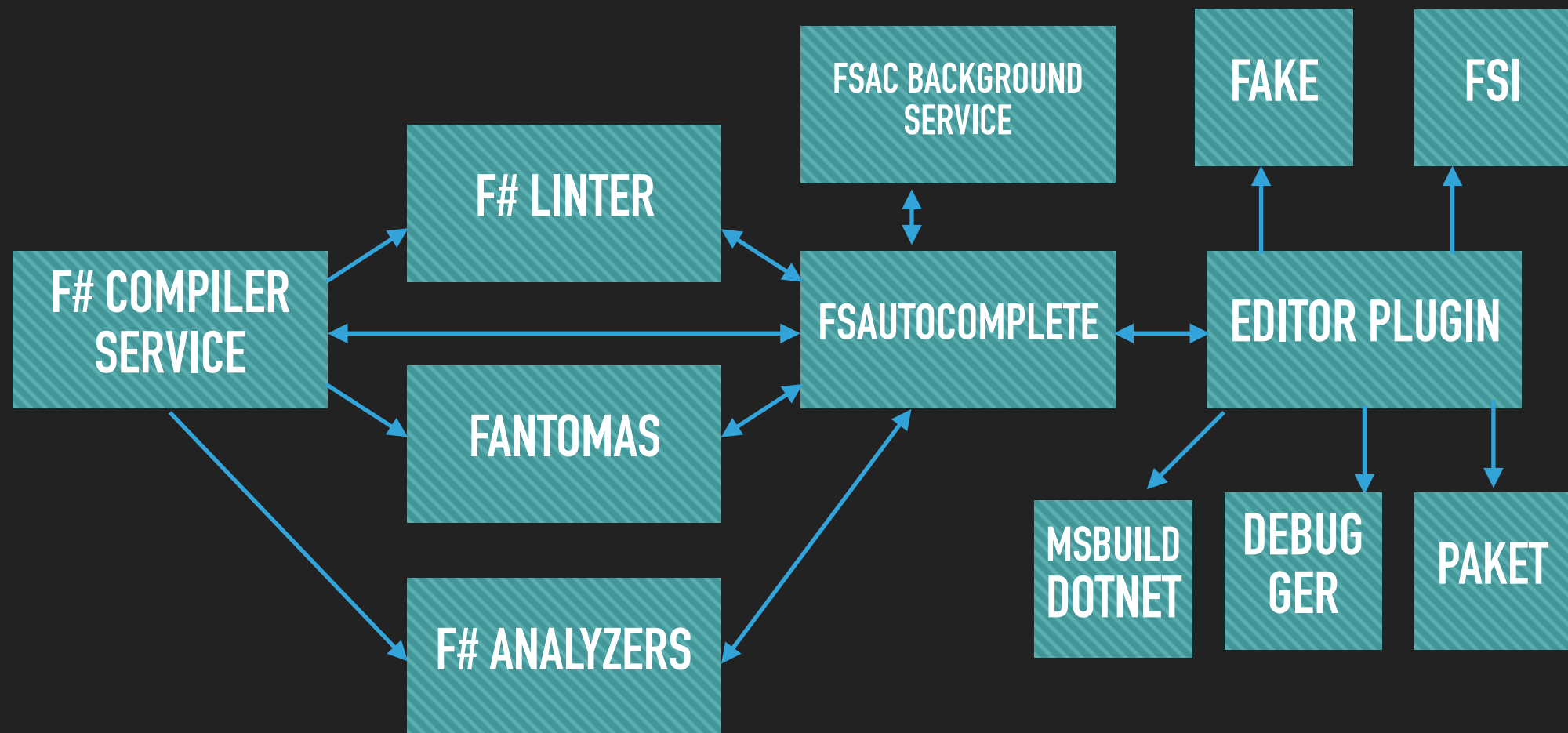
LANGUAGE SERVER FOR F#

HIGH LEVEL API

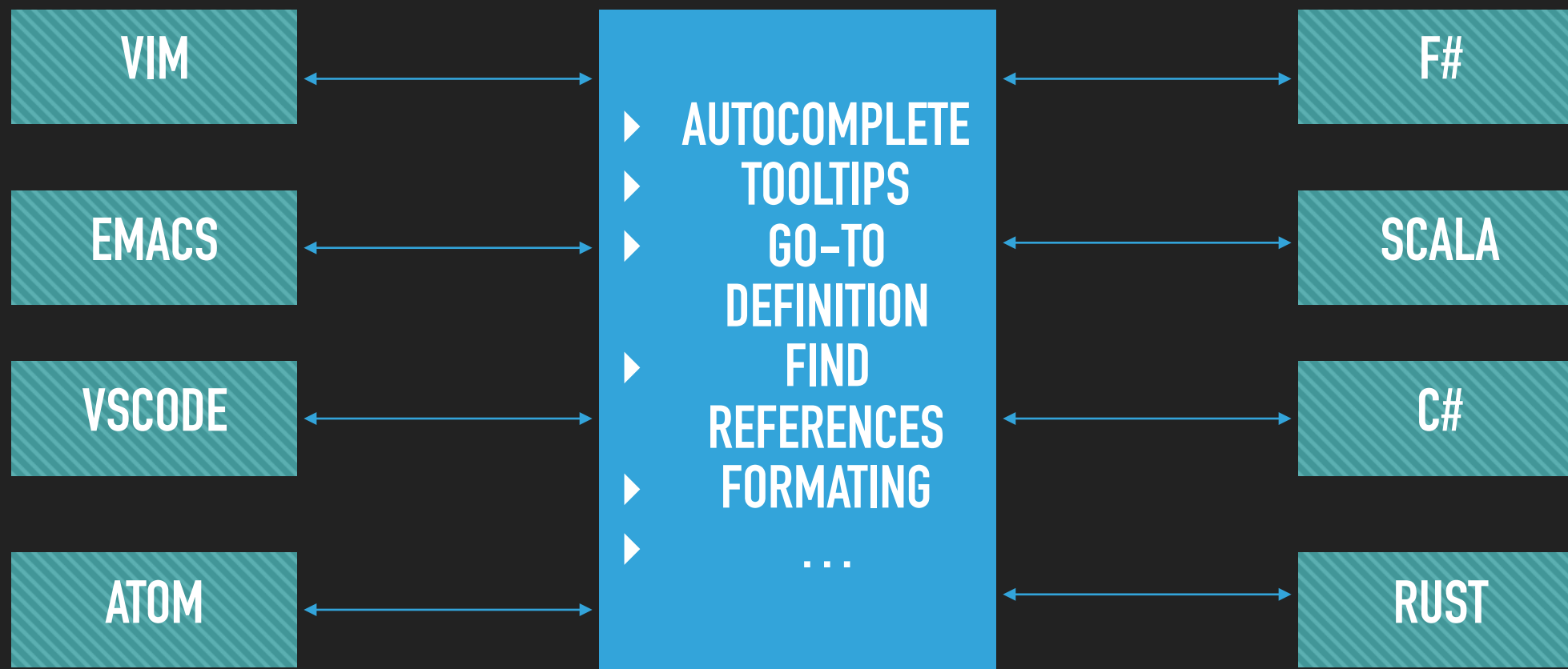
NOWADAYS USING LSP
(THANKS MSFT)

FSAUTOCOMPLETE

EDITOR TOOLING ARCHITECTURE



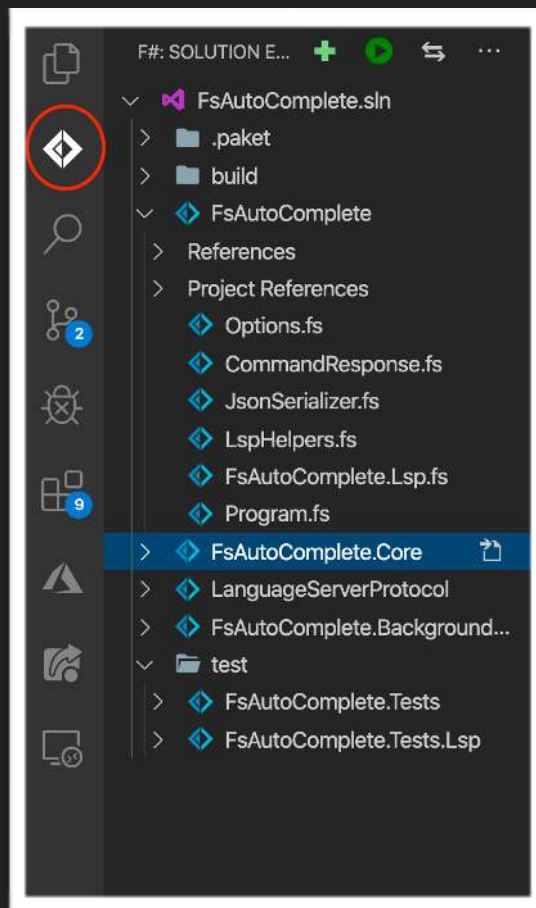
LANGUAGE SERVER PROTOCOL



**LSP IS NOT ENOUGH TO
BUILD POWERFUL IDE**

LANGUAGE SERVER PROTOCOL

EXPANDING LSP



```
Seq.iter (fun path →  
    match state.FileCheckOptions.TryRemove path with  
    | true,  
    val iter : action:('T -> unit) -> source:seq<'T> -> unit
```

Info Panel

```
val iter:  
  action: 'T -> unit ->  
  source: source  
  -> unit
```

Description

Applies the given function to each element of the collection.

Parameters

- action:** A function to apply to each element of the sequence.
- source:** The input sequence.

Exceptions

- System.ArgumentNullException:** Thrown when the input sequence is null.

Generic parameters

- 'T** is **SourceFilePath**

Full name: *Microsoft.FSharp.Collections.Seq.iter*

Declaring Entity: [Seq](#)

Assembly: *FSharp.Core*

```
let mutable lastVersionChecked = -1 // int  
  
let mutable lastCheckResult : ParseAndCheckResults option = None // ParseAndCheckResults option  
  
let mutable isWorkspaceReady = false // bool  
  
let notify = Event<NotificationEvent>() // Event<NotificationEvent>
```




DISTRIBUTE CUSTOM
ANALYSERS AS NORMAL
PACKAGES

CUSTOM WARNINGS,
ERRORS AND CODE FIXES

INTEGRATED WITH
TOOLING

F# ANALYZERS

F# ANALYZERS

1 Ref
let
[!En
0 Ref
let

Option.Value analyzer: Option.Value shouldn't be used
property Value: unit
Full name: Microsoft.FSharp.Core.Option.Value
Assembly: FSharp.Core
x.Value
printfn "Hello World from F#!"

Program.fs X

sample > FSharp.EventHorizon.Sample > Program.fs > {} Program > main

1
2 open System
3 open EventHorizon.Hole
4
0 References | int list -> int
5 let mySum (args: int list) = List.sum args
6
0 References | int list -> int
7 let otherSum = fun (args: int list) -> List.sum args
8
0 References | 'a list -> 'a list
9 let inline myMap x = List.filter (fun a -> true) x
10
0 References | int list -> int list
11 let myMap2 = List.map (fun x -> x + 1)
12
0 References | 'a list -> int list
13 let myMap3 x = List.map (fun x -> 1) x
14
15
16 [0 References | string [] -> int
17 let main argv =
18 let result =
19 [1 .. 10]
20 > List.map (fun i -> i * i)
21 > hole
22 >
23 printfn
24 0 // re
25

**LET'S TALK ABOUT
DEVELOPER EXPERIENCE**



RICH

UNINTRUSIVE

CORRECT

CONTEXT-AWARE

RESPONSIVE

PERFORMANT

**MODERN EDITOR
TOOLING**

**BUILDING TOOLING
IS ART OF BALANCE**



EVERYONE HAS
THEIR
PREFERRED
WORKFLOW

**FOCUS ON FINDING
GOOD DEFAULTS**



FOCUS ON GETTING STARTED
EXPERIENCE

TOOLS SHOULD “JUST WORK”

**USERS DON'T
READ README**

**USER EXPERIENCE OVER
TECHNICAL PERFECTION**



RELEASE OFTEN EVEN IF IT'S NOT
PERFECT

GETTING FEEDBACK FROM AS MANY
USERS AS POSSIBLE

VALUE OF SHORT FEEDBACK LOOP



MAKE SURE IT DOESN'T
LEAK PROBLEMS OF
LOWER LAYERS

**TOOLING IS AN
ABSTRACTION LAYER**

**BUILDING TOOLING
IS FUN, TRY IT OUT!**



- ▶ COMPILER DESIGN
 - ▶ COMPILER AS A SERVICE
 - ▶ LANGUAGE SERVER PROTOCOL
 - ▶ USER EXPERIENCE
 - ▶ F# TOOLING ARCHITECTURE
-

**WHAT I'VE
TALKED ABOUT**

**ASK ME
ANYTHING**



@K_CIESLAK

KRZYSZTOF_CIESLAK@WINDOWS.LIVE.COM

GITHUB.COM/KRZYSZTOF-CIESLAK

500PX.COM/KRZYSZTOF-CIESLAK

GOODREADS.COM/KRZYSZTOF-CIESLAK

LAMBDAFACTORY.IO

OPENCOLLECTIVE.COM/IONIDE

**KRZYSZTOF
CIESLAK**