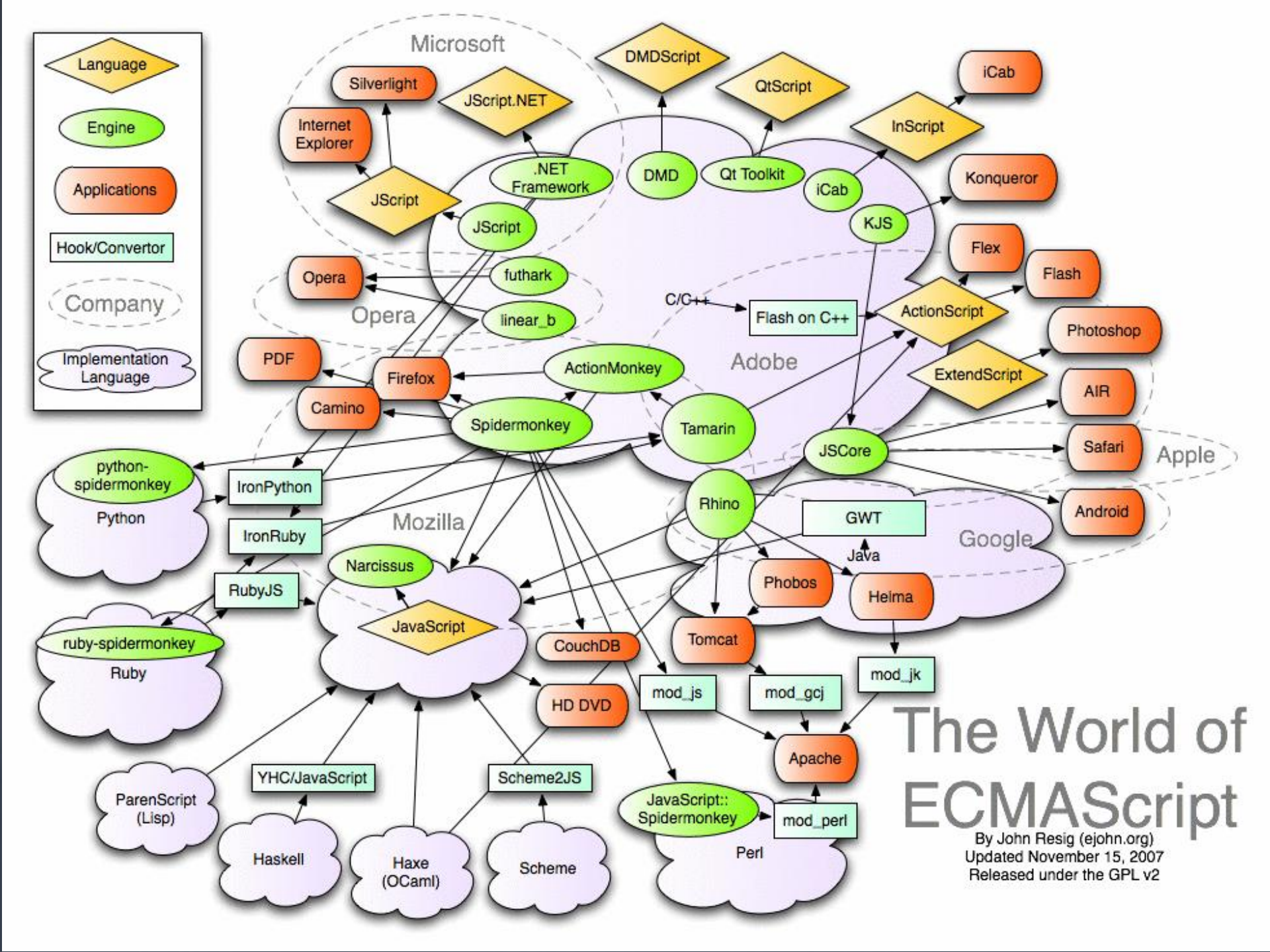
j's way.

{ JavaScript语言的进化和选择

周爱民/aimingoo
http://blog.csdn.net/aimingoo
aiming@gmail.com


Language
Engine
Applications
Hook/Convertor
Company
Implementation Language
Microsoft
Silverlight
JScript.NET
Internet Explorer
JScript
.NET Framework
JScript
DMDScript
DMD
QtScript
Qt Toolkit
iCab
InScript
iCab
Konqueror
KJS
Opera
Opera
futhark
linear_b
PDF
Firefox
Camino
ActionMonkey
Spidermonkey
Tamarin
C/C++
Flash on C++
Adobe
Flex
Flash
ActionScript
Photoshop
ExtendScript
AIR
JSCore
Safari
Apple
Android
Rhino
GWT
Java
Google
Phobos
Helma
Tomcat
mod_jk
mod_gcj
mod_js
Apache
mod_perl
JavaScript:: Spidermonkey
Perl
python-spidermonkey
Python
IronPython
IronRuby
Mozilla
RubyJS
ruby-spidermonkey
Ruby
Narcissus
JavaScript
CouchDB
HD DVD
ParenScript (Lisp)
YHC/JavaScript
Haskell
Haxe (OCaml)
Scheme2JS
Scheme
The World of ECMAScript
By John Resig (ejohn.org)
Updated November 15, 2007
Released under the GPL v2

JavaScript

Development Tools

IDE : Eclipse, Aptana, Dreamweaver, Spket
Debug : Firebug, Web Inspector,
Optimizer : YUI Compress, JSMin, JSLint, Google Closure

Ajax with Web2.0

User need more better User Interface, User Expreience

Browsers Competition

FireFox, Chrome, Safari, Opera, IE

Web Standard Evolution

HTML5, CSS3, WebGL, Web Application API, XHR

Frameworks & Open Sources

Prototype, jQuery, YUI, Dojo, ExtJS, Mootools, MethodChain ...

for Desktop Application

Adobe AIR , Chrome OS

Open API for Data Portability

facebook, myspace, twitter, youtube, flickr ...

for Mobile

iPhone, Nokia, Android,
Window Mobile

for Server side

Jaxer

语言特性

- OOP
- functional
- dynamic

History

v1.0, NN2

Structured

Object & DOM…

Function/Method/Event

eval

v1.1, NN3

base prototype, constructor

functonal, anonymous

typeof, void, delete

Object & DOM ++

v1.2, NC4.0

regular expressions

signed scripts

labeled statements

switch statements

v1.3, NC4.07

NaN, Infinity, undefined

call, apply

===, !==

Literal: Array, Object, Primitive Values

ECMA. with out DOM.

Object System

- 到底什么是（面向）对象系统？
- JS1.0如何理解对象系统？
- 从原型继承开始，面向对象三个特征的梳理。
- 方向：从原型继承到类继承

History

v1.4, Live

throw and try-catch

in, instanceof

- foo.arguments

- foo.arity

v1.5, NS6

Conditional function

Function expressions

Multiple catch clauses

Const, Getters and Setters

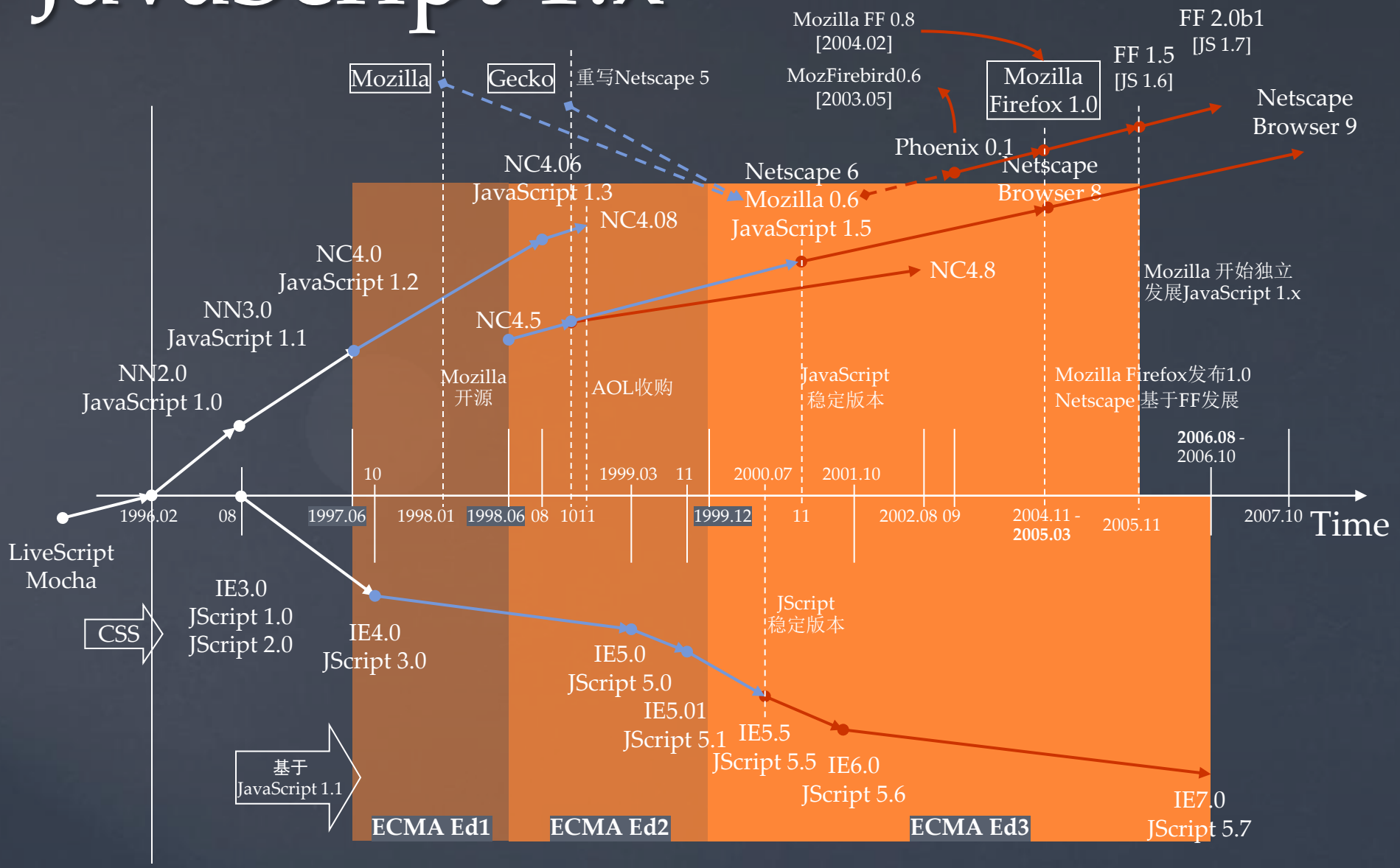
History

- window.eval/execScript
- object.eval
- call/apply/with
- ‘use static’

dynamic, functional

- 动态绑定是不是好的特性？
- 脚本与动态执行
- 动态执行带来的系统负担 - Closure compiler
- 函数式特性主要出自于对语言的完善
- 三种闭包域，以及语法作用域的完整性问题
- 方向：将动态特性限制在一定范围内(es5)

JavaScript 1.x



JavaScript 1.4只由Netscape服务器端产品实现

JScript 4.0只包含在Visual Studio早期的SP中

语法

- 语法简洁
- 结构化

History

v1.6, FF1.5

E4X

Array.forEach()…

for each (v in o) …

v1.7, FF2.0 b1

yield <g>, g.next/throw/close()

Iterator(o)对o.__iterator__属性的重载

数组领悟(for each在数组声明中的使用)

let及其作用域

解构赋值, 多值返回(在for中的应用)

v1.8 FF3, Gecko 1.9

表达式增强

```
function(x) x * x
it = (i + 3 for (i in someObj));
```

数组增强(reduce, reduceRight)

v1.81, FF3.5, Gecko 1.91, SeaMonkey 2 *(base Tracemonkey)*

Object.getPrototypeOf()

native JSON

String.trim/trimLeft/trimRight()

语法增强

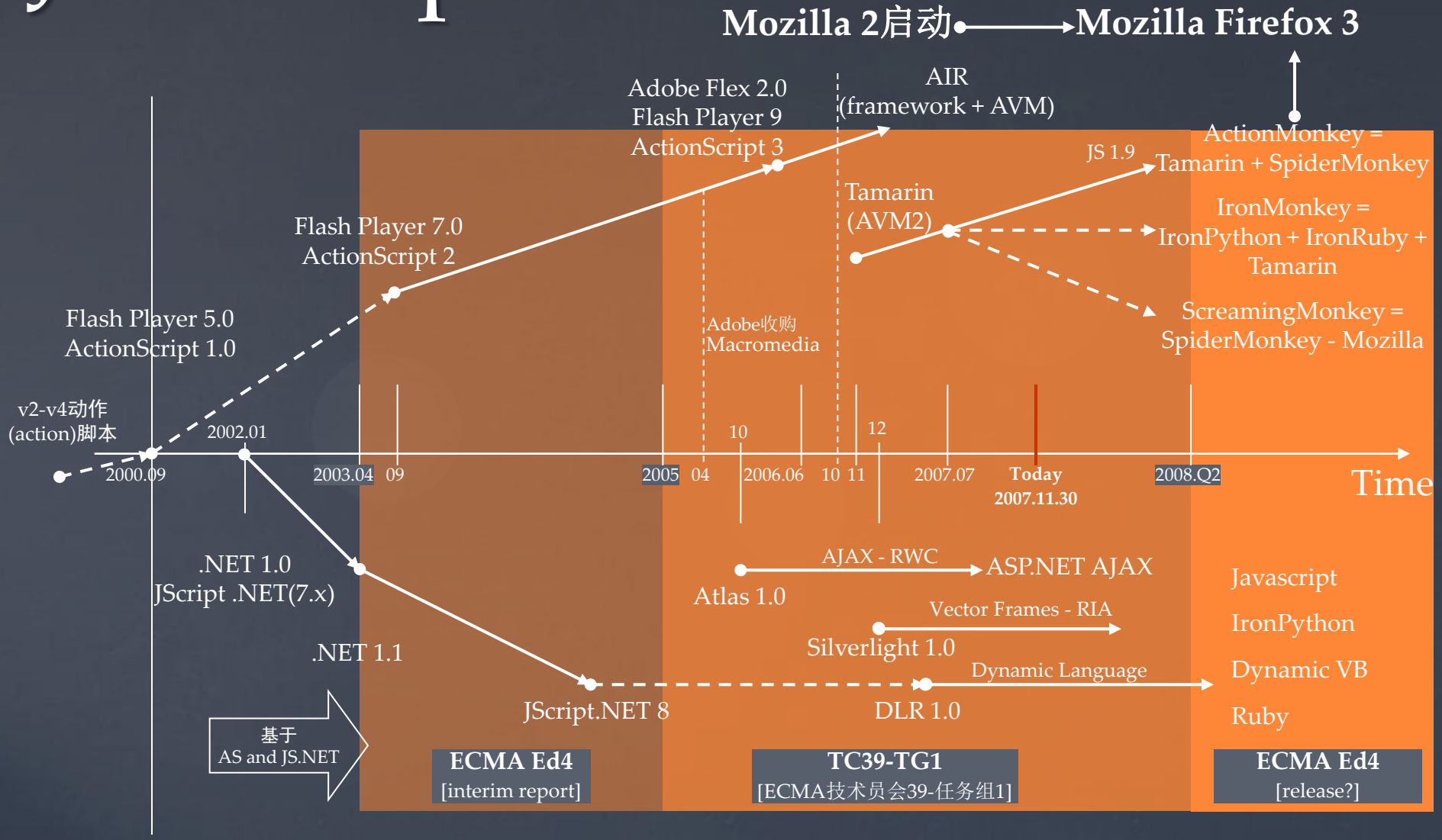
- 结构化语法特性的增强
- 作用域的补充
- (面向函数式语言的)表达式的增强
- 面向系统的结构化增强
  - 包、命名空间与注解
  - program, package, class, function
  - namespace, use
- 方向：必要性与灵活的平衡

面向对象语法的增强

- class MyClass extends BaseClass () {
- property name1;
- include BaseClass2;
- static function name2 () { ... }
- }
- var num:Number = 10;

- as, is, super, static, ...

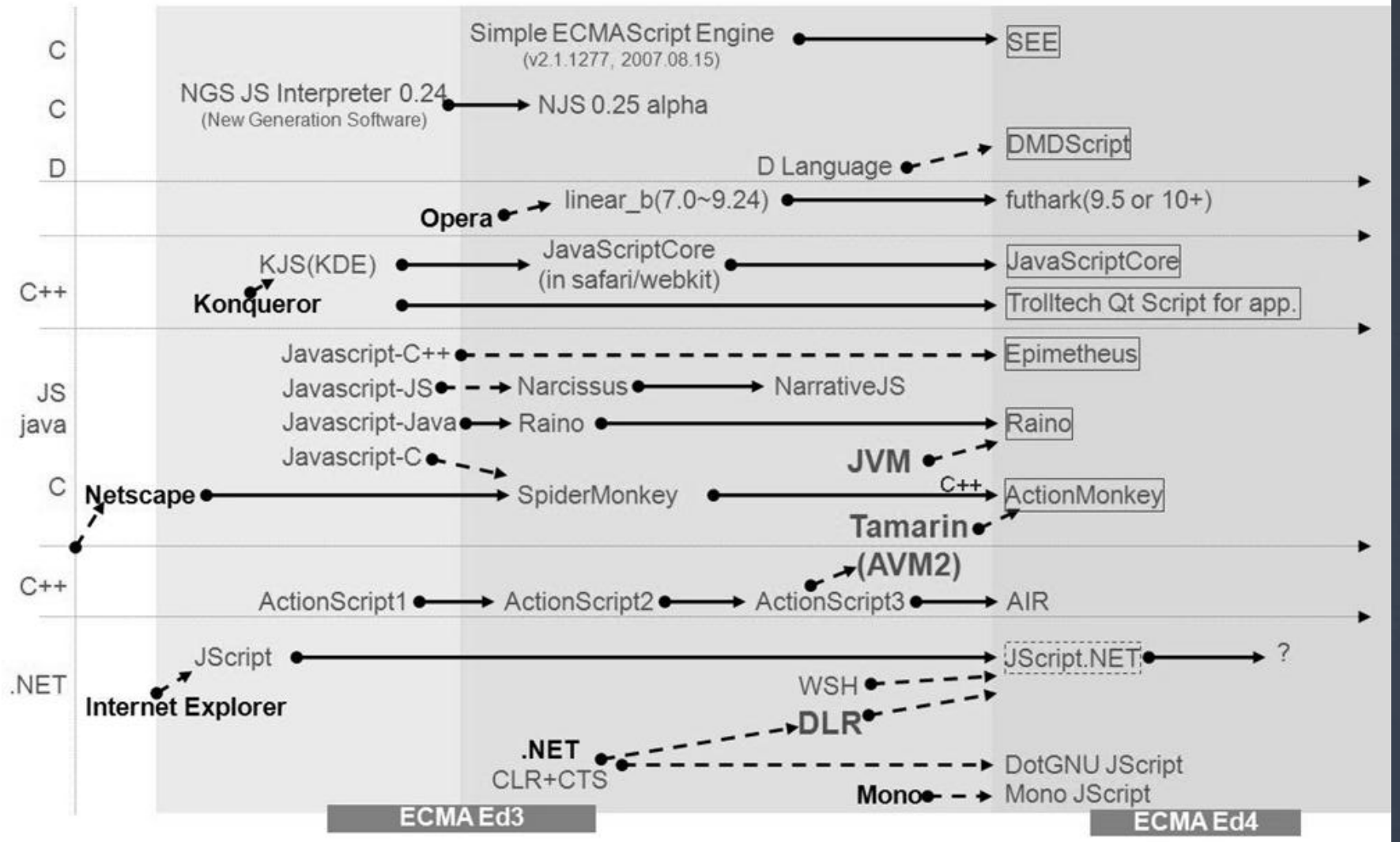
JavaScript 2



ES3: 对大型软件系统的抽象能力是较弱的，也缺乏一些大型编程系统中常用的机制（例如静态类型检查、早期绑定等）此外，它是基于对象而非面向对象的，因此对象系统的表达能力也不足够。

AS3: 使用双虚拟机支持AS1-2与AS3，基本重写语言框架。

JavaScript family



ECMA Ed5.

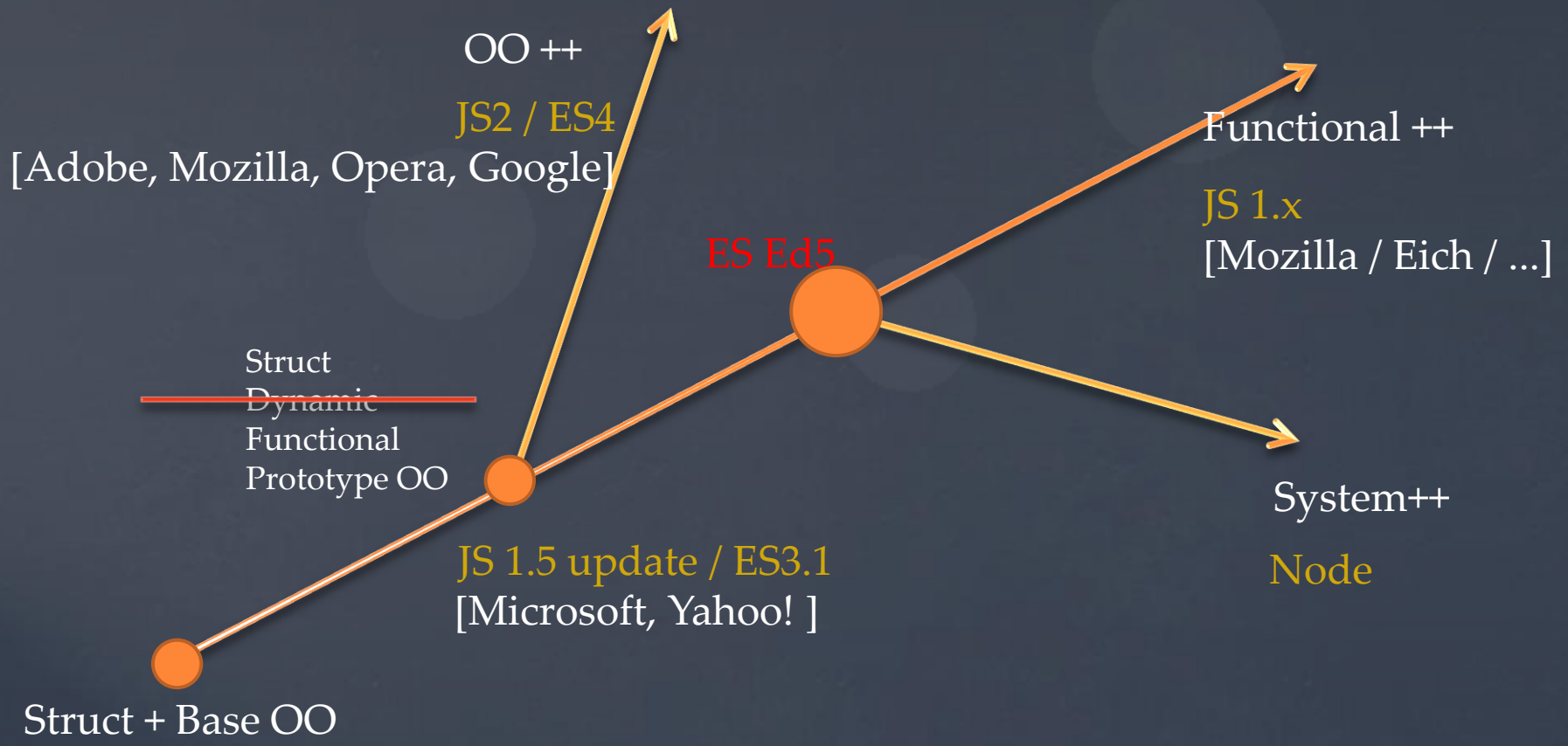
- 在Meta级别上进行OOP扩展的尝试；
- 安全、高效的函数式语言特性扩展；
- 严格（strict）模式下的函数式语言特性。
- “无类语言”与对象系统的扩展与限制

分类	方法	说明
属性声明 (*注)	create(prototypeObj, props)	使用 prototypeObj 为原型来构造对象，并声明一组属性
	defineProperty(obj, name, desc)	为对象声明一个属性
	defineProperties(obj, props)	为对象声明一组属性
	getOwnPropertyDescriptor(obj)	取对象的属性描述符列表(props)
取属性列表	getOwnPropertyNames(obj)	取对象自有的属性名数组
	keys(obj)	取对象所有的(包括继承来的)属性名数组
属性状态维护	preventExtensions	使实例 obj 不能添加新属性
	seal(obj)	使实例 obj 不能添加新属性，也不能删除既有属性
	freeze(obj)	使实例 obj 所有属性只读，并且不能再添加、删除属性
	isExtensible	返回 preventExtensions 状态
	isSealed	返回 seal 状态
	isFrozen	返回 freeze 状态

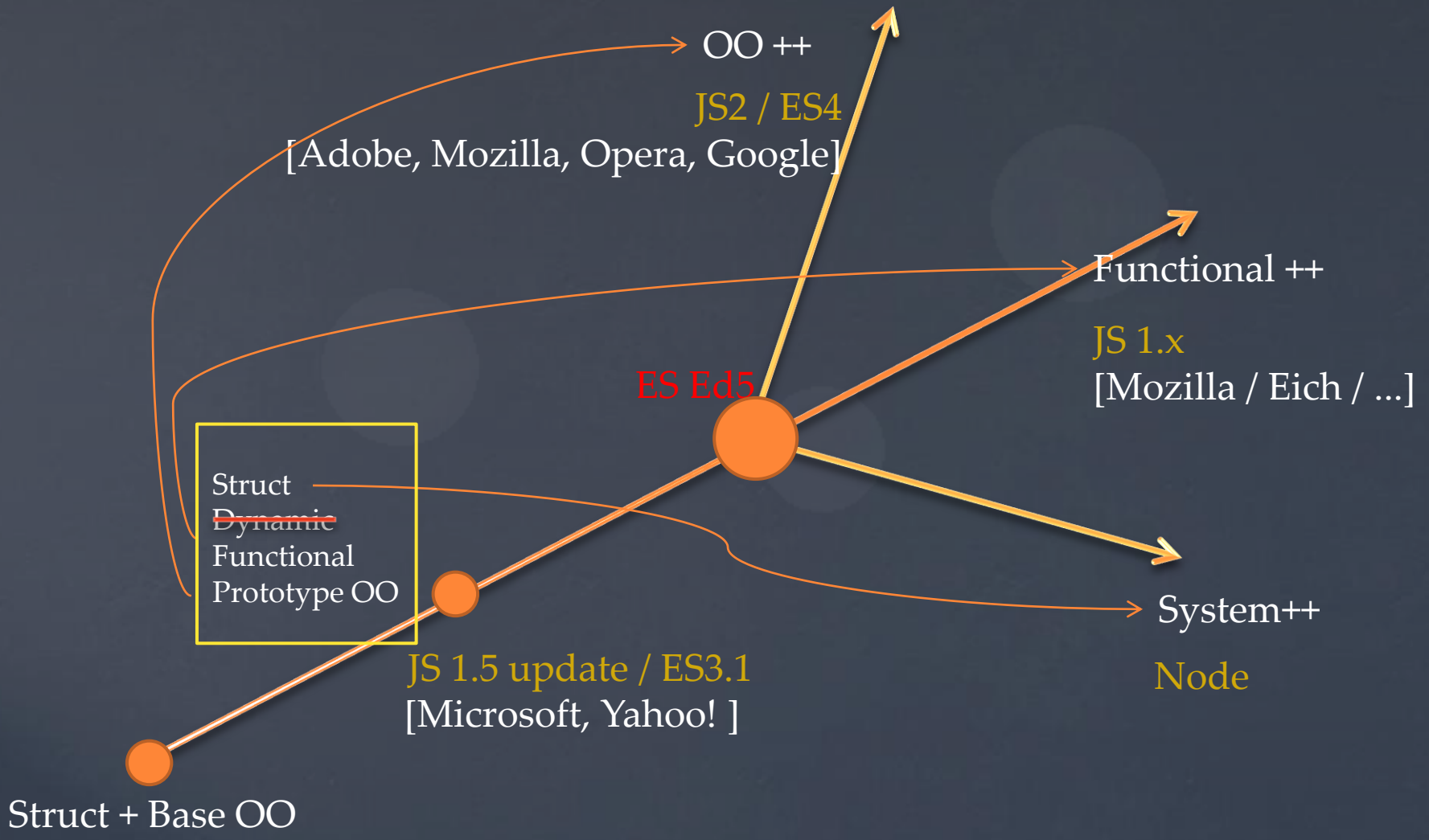
- ECMA Ed5.
- ECMAScript Harmony

标准化、Coffee与paren-free

Crossing - 4种语言特性的博弈



Crossing - 4种语言特性的博弈



标准化

- **ECMAScript Harmony.**

	ES4 RI	Tamarin	Spidermonkey	Rhino	Futhark	Mbedthis
Types	107	44	29	21	12	91
Classes	15	7	0	0	0	12
Structure	7	5	1	1	1	10
Declarations	5	1	3	3	2	3
Statements	2	4	3	3	0	7
Operators	45	25	8	8	3	48
Iteration	7	0	2	0	0	8
Functions	12	2	1	0	0	4
Objects	2	0	1	1	0	3
RegExp	3	1	1	0	2	0
String	3	2	3	2	2	2
Array	10	7	11	8	1	8
Builtins	4	0	0	0	1	3
Misc	0	0	0	0	0	4

编译、VM与某些方向

- **CoffeeScript 1.0**
- **StratifiedJS**
- Emscripten
- Node & CommonJS
- Closure Compiler

paren-free

- **Eich's dreams.**

```
if year > 2010 {
   syntax++
}
```

paren-free

```
function add(a, b) { return a + b }
(function(x) { return x * x })
```

```
const #add(a, b) { a + b }
#(x) { x * x }
```

paren-free

```
(function () {
  return typeof arguments
})() == "undefined"
```

```
#() {
  typeof arguments
}() == "undefined"
```

```
var point = {x: 10, y: 20}
point.equals({x: 10, y: 20})
```

```
const point = #{x: 10, y: 20}
point === #{x: 10, y: 20}
```

paren-free

```
var tuple = [1, 2, 3]
tuple[tuple.length-1] === 3
Array.prototype.compare = /*...*/
tuple.slice(0, 2).compare([1, 2]) == 0
tuple.compare([1, 2, 4]) < 0
```

```
const tuple = #[1, 2, 3]
tuple[-1] === 3

tuple[0:2] === #[1, 2]
tuple < #[1, 2, 4]
```

paren-free

```
if (x > y) alert("brace-free")
if (x > z) return "paren-full"
if (x > y) f() else if (x > z) g()
```

```
if x > y { alert("paren-free") }
if x > z return "brace-free"
if x > y { f() } else if x > z {
```

have prototyped in Narcissus (invoked via njs --paren-free)

paren-free

```
function construct(f, a) {
  switch (a.length) {
    case 0: return new f
    case 1: return new f(a[0])
    case 2: return new f(a[0], a[1])
    default:
      var s = "new f("
      for (var i = 0; i < a.length; i++)
       s += "a[" + i + "],"
     s = s.slice(0, -1) + ")"
     return eval(s)
  }
}
```

```
function construct(f, a) {
  return new f(...a)
}
```

{ END.