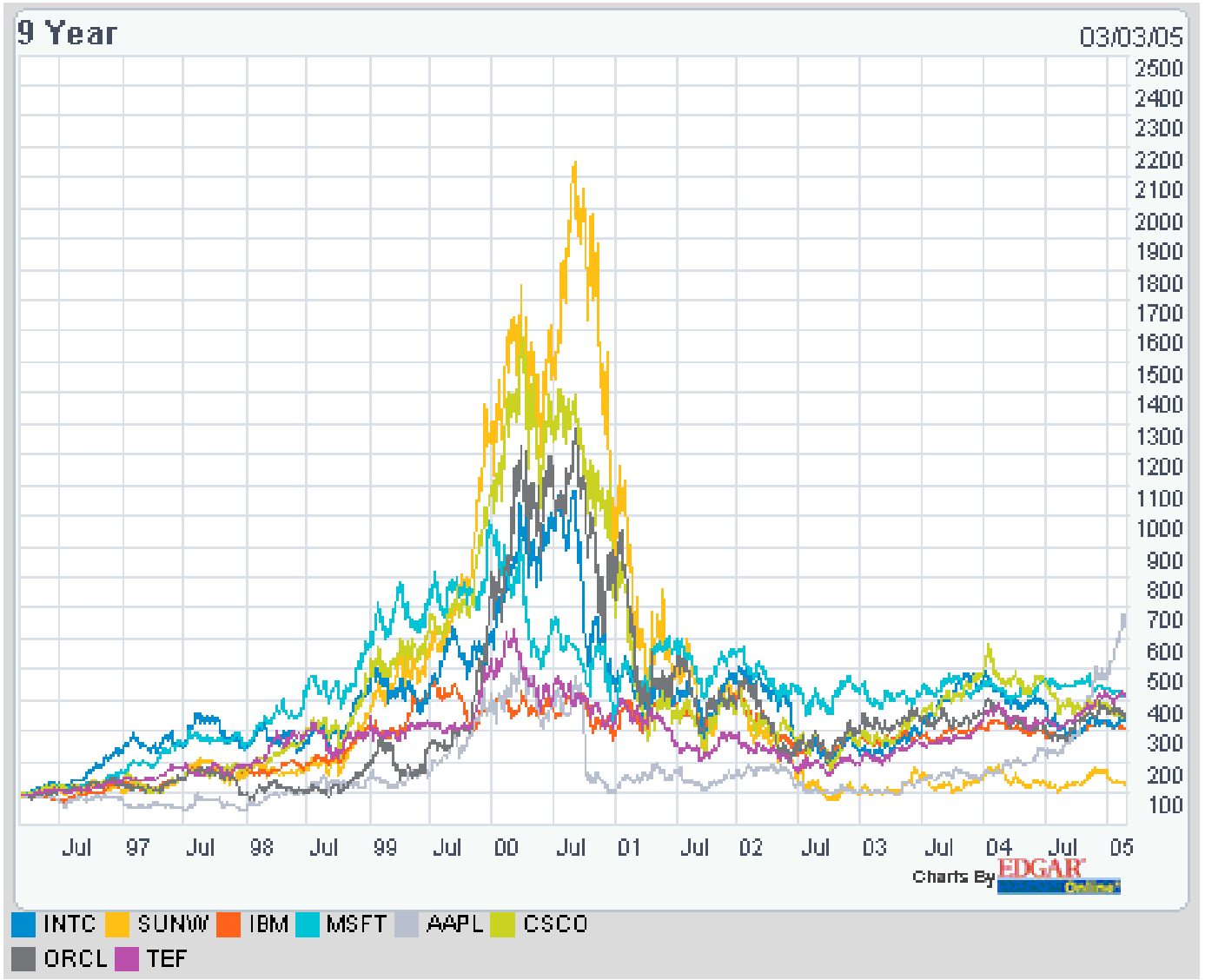
TAP. ¿ES JAVA LA MEJOR OPCIÓN?

onente importante del "boom" de las "dotcom" fue Java



Sun fue cada vez mejor representada por Java hasta el punto de llegar a cambiar su "ticker" en bolsa

siliconnews.es
La actualidad empresarial y de las nuevas tecnologías

Negocios y Mercados

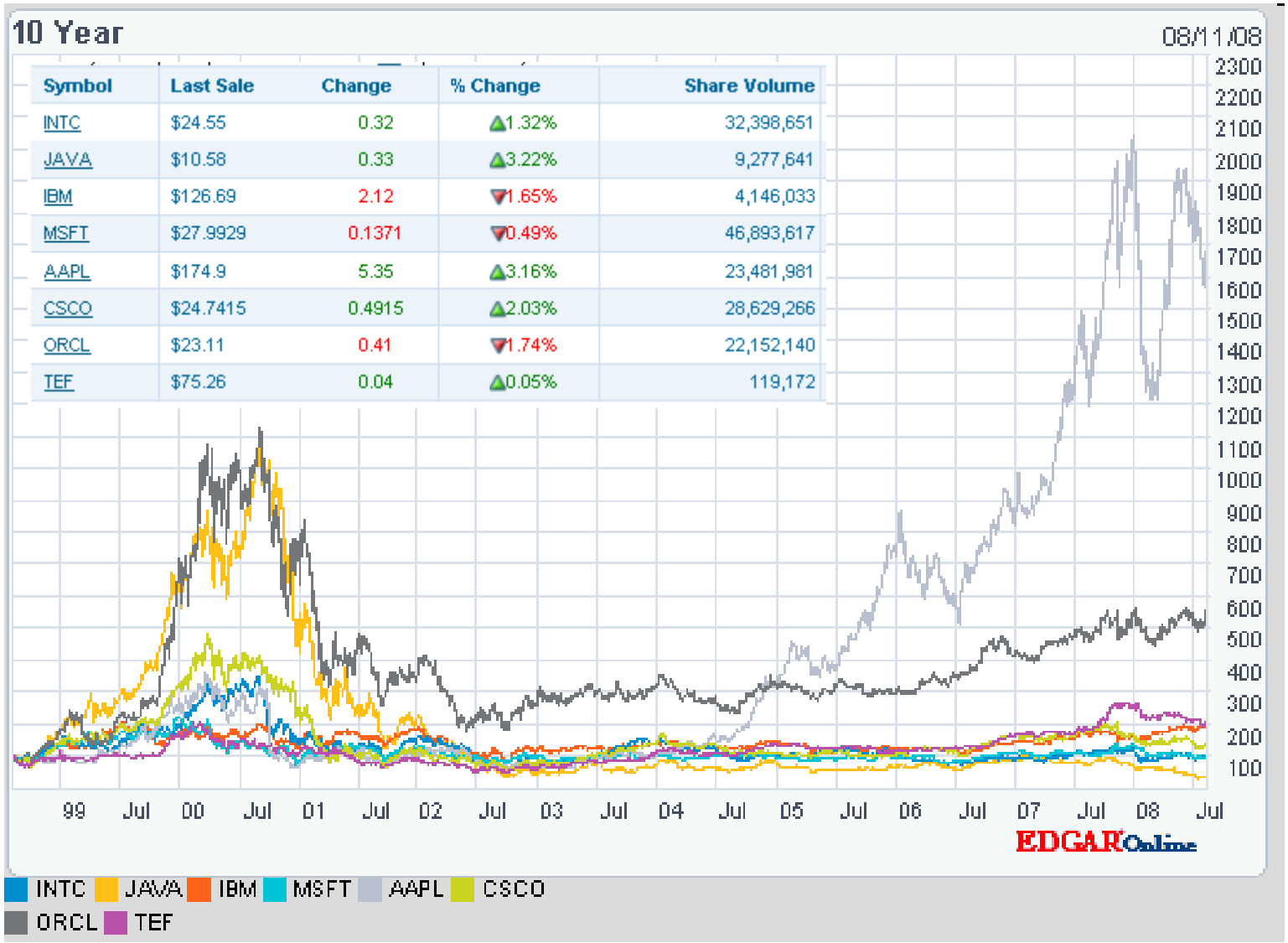
Sun Microsystems pasará a ser Java en el Nasdaq

25 ago 07 | 11:25 CET

A partir del 27 de agosto, la compañía tecnológica cambiará su actual identificación bursátil, SUNW, por Java, su marca más reconocida en el mercado.

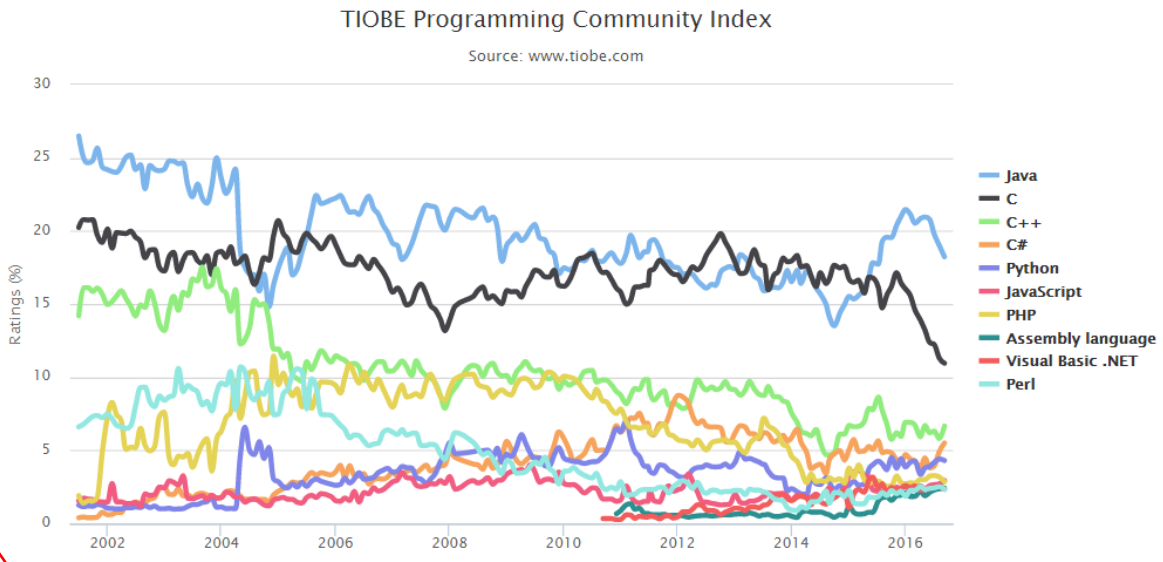
Desde el próximo 27 de agosto, la identificación bursátil de la compañía tecnológica Sun Microsystems dejará de ser SUNW para pasar a ser **Java, su marca más reconocida en el mundo del software**.
En declaraciones reproducidas por Europa Press, el presidente y consejero delegado de la empresa, Jonathan Schwartz, aseguró que esta modificación de las siglas de identificación bursátil "refleja una marca que todo el mercado puede identificar y supone un elemento importante del proceso de transformación de Sun a largo plazo".
El directivo agregó que "Java está en todas partes, tocando de cerca a cualquiera que esté relacionado con Internet y es un símbolo de la capacidad de desarrollar, introducir y dar a conocer las novedades de Sun".
Cabe recordar que, según un informe de Ovum basado en estadísticas de mayo de este año, existen **800 millones de ordenadores con software Java incorporado, 2.100 millones de dispositivos móviles para Java, 2.500 millones de tarjetas inteligentes y cerca de 180 operadores que ofrecen contenidos y servicios basados en esta tecnología.**

dad: Apple "relativiza" el boom de las "dotcom"



ación de lenguajes por presencia en la web

Sep 2016	Sep 2015	Change	Programming Language	Ratings	Change
1	1		Java	18.236%	-1.33%
2	2		C	10.955%	-4.67%
3	3		C++	6.657%	-0.13%
4	4		C#	5.493%	+0.58%
5	5		Python	4.302%	+0.64%
6	7	^	JavaScript	2.929%	+0.59%
7	6	v	PHP	2.847%	+0.32%
8	11	^	Assembly language	2.417%	+0.61%
9	8	v	Visual Basic .NET	2.343%	+0.28%
10	9	v	Perl	2.333%	+0.43%
11	13	^	Delphi/Object Pascal	2.169%	+0.42%
12	12		Ruby	1.965%	+0.18%
13	16	^	Swift	1.930%	+0.74%
14	10	vv	Objective-C	1.849%	+0.03%
15	17	^	MATLAB	1.826%	+0.65%
16	34	^^	Groovy	1.818%	+1.31%
17	14	v	Visual Basic	1.761%	+0.23%
18	19	^	R	1.684%	+0.64%
19	44	^^	Go	1.625%	+1.37%
20	18	v	PL/SQL	1.443%	+0.36%



Position	Programming Language	Ratings
21	SAS	1.267%
22	ABAP	1.167%
23	Dart	1.089%
24	COBOL	1.064%
25	D	0.939%
26	Scratch	0.840%
27	Fortran	0.683%
28	Lisp	0.641%
29	F#	0.605%
30	Lua	0.586%
31	Ada	0.567%
32	Scala	0.549%
33	Transact-SQL	0.520%
34	Prolog	0.450%
35	LabVIEW	0.450%
36	Logo	0.410%
37	Apex	0.355%
38	Q	0.334%
39	Scheme	0.311%
40	Haskell	0.304%
41	RPG (OS/400)	0.301%
42	Erlang	0.278%
43	Bash	0.269%
44	Ladder Logic	0.266%
45	Rust	0.258%
46	Awk	0.209%
47	Julia	0.196%
48	Alice	0.193%
49	VHDL	0.192%
50	Clojure	0.166%

http://www.tiobe.com/tiobe-index/

¿QUÉ “CLASE” DE LENGUAJE ES JAVA?

La Programación ES y VIENE DE las Matemáticas… (como parte de la computación)

Euclides (Método axiomático), Aristóteles(Lógica formal), Muhammad ibn Musa Al'Khowarizmi (Algorítmo)…

David Hilbert (Frege Russel Whitehead)
Kurt Gödel

1931
1938

Máquina de Turing

Alan Turing

Cálculo Lambda

Sin olvidar las funciones recursivas de Herbrand-Gödel"

Alonzo Church

Lenguajes

Imperativos
- Fortran
- Cobol
- Pascal
- C
- …

O/B objetos
- Object Pascal
- C++
- Javascript
- Java
- …

Frameworks
- Ruby on Rails
- …

Funcionales
- Lisp
- Scheme
- ML
- Hope
- CLOS
- Ocaml
- …
- Haskell
- Clojure

Lógicos
- Prolog
- …

- Javascript
- Scala
- Java 8

! no hay asignación, (todo es inmutable) !

```
let rec long = function
      |[] -> 0
      |x::xs -> 1 + long xs;;

let rec ordenar = function
      |[] -> []
      |x::xs -> insertar x (ordenar xs)
and insertar e = function
      |[] -> [e]
      |x::xs -> if x > e
            then e::x::xs
            else x::(insertar e xs);;
```

Ejemplo OCaml

Alan **Turing** - Wikipedia, la enciclopedia libre

es.wikipedia.org/wiki/Alan_**Turing**

Alan Mathison **Turing**, OBE (Paddington, Londres, 23 de junio de 1912 - Wilmslow, Cheshire, 7 de junio de 1954), fue un matemático, lógico, científico de la ...

Máquina de Turing

Una máquina de Turing es un dispositivo que manipula ...

Tesis de Church-Turing

En teoría de la computabilidad, la tesis de Church-Turing formula ...

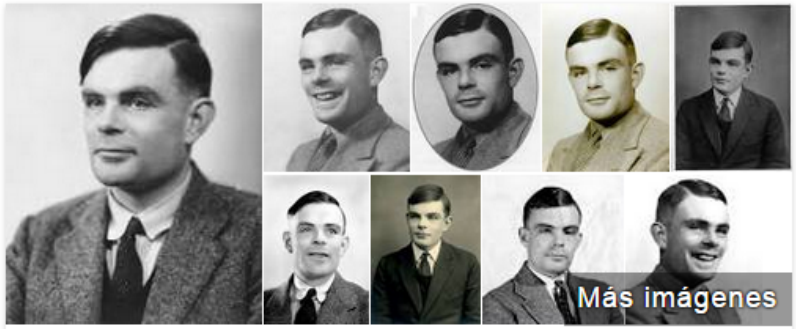
Enigma

Enigma era el nombre de una máquina que disponía de un ...

Problema de la parada

El problema de la parada o problema de la detención para ...

Más resultados de wikipedia.org »



Alan Turing

Matemático

Alan Mathison Turing, OBE, fue un matemático, lógico, científico de la computación, criptógrafo y filósofo británico. Es considerado uno de los padres de la ciencia de la computación siendo el precursor de la informática moderna. Wikipedia

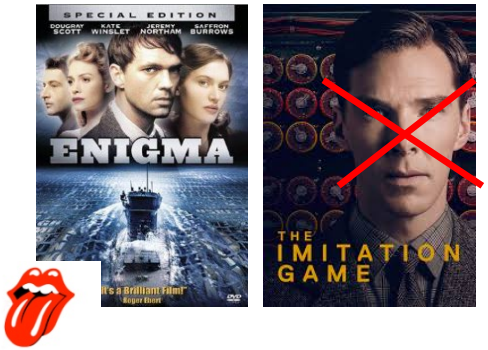
Fecha de nacimiento: 23 de junio de 1912, Maida Vale, Londres, Reino Unido

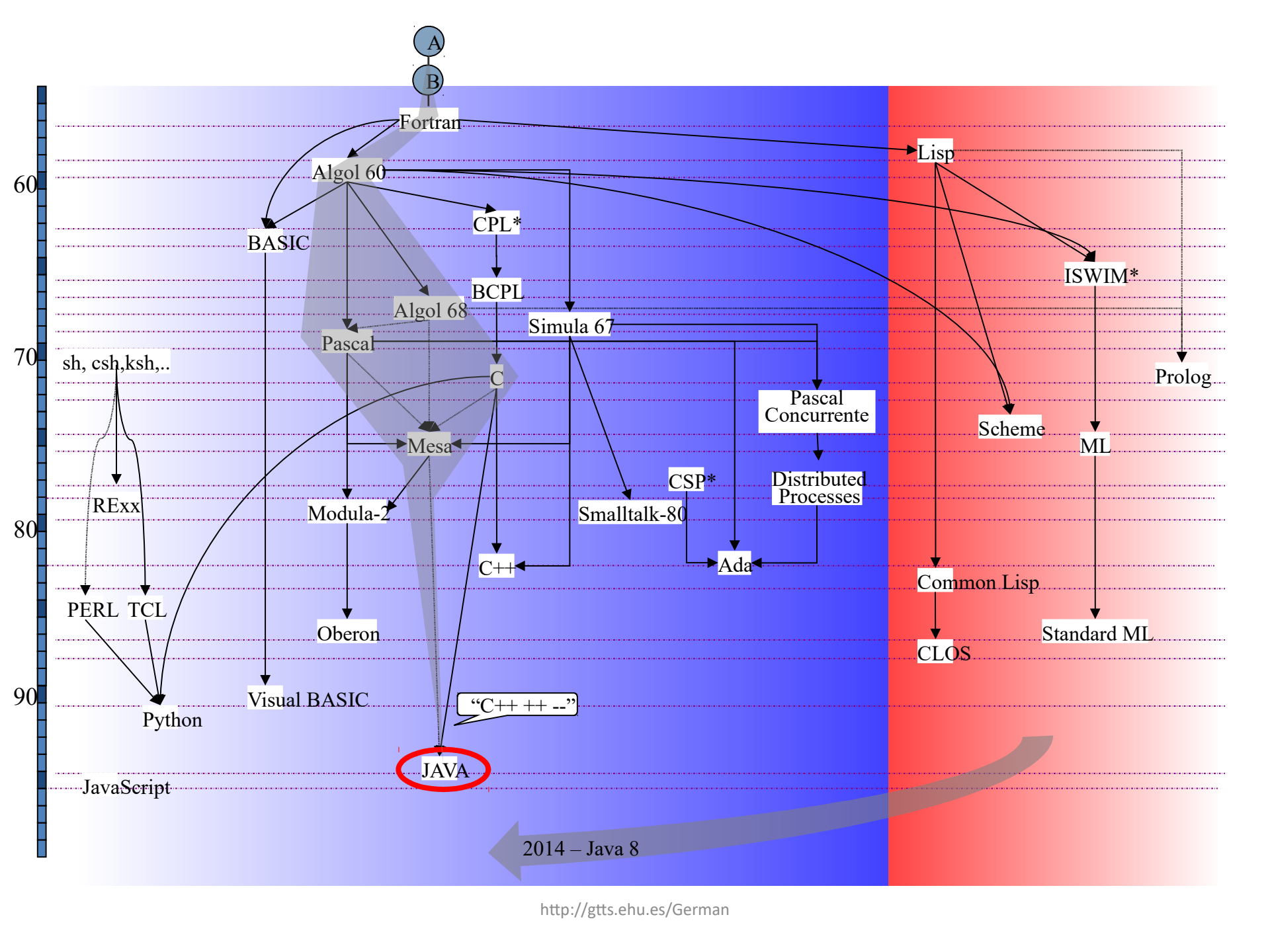
Fecha de la muerte: 7 de junio de 1954, Wilmslow, Reino Unido

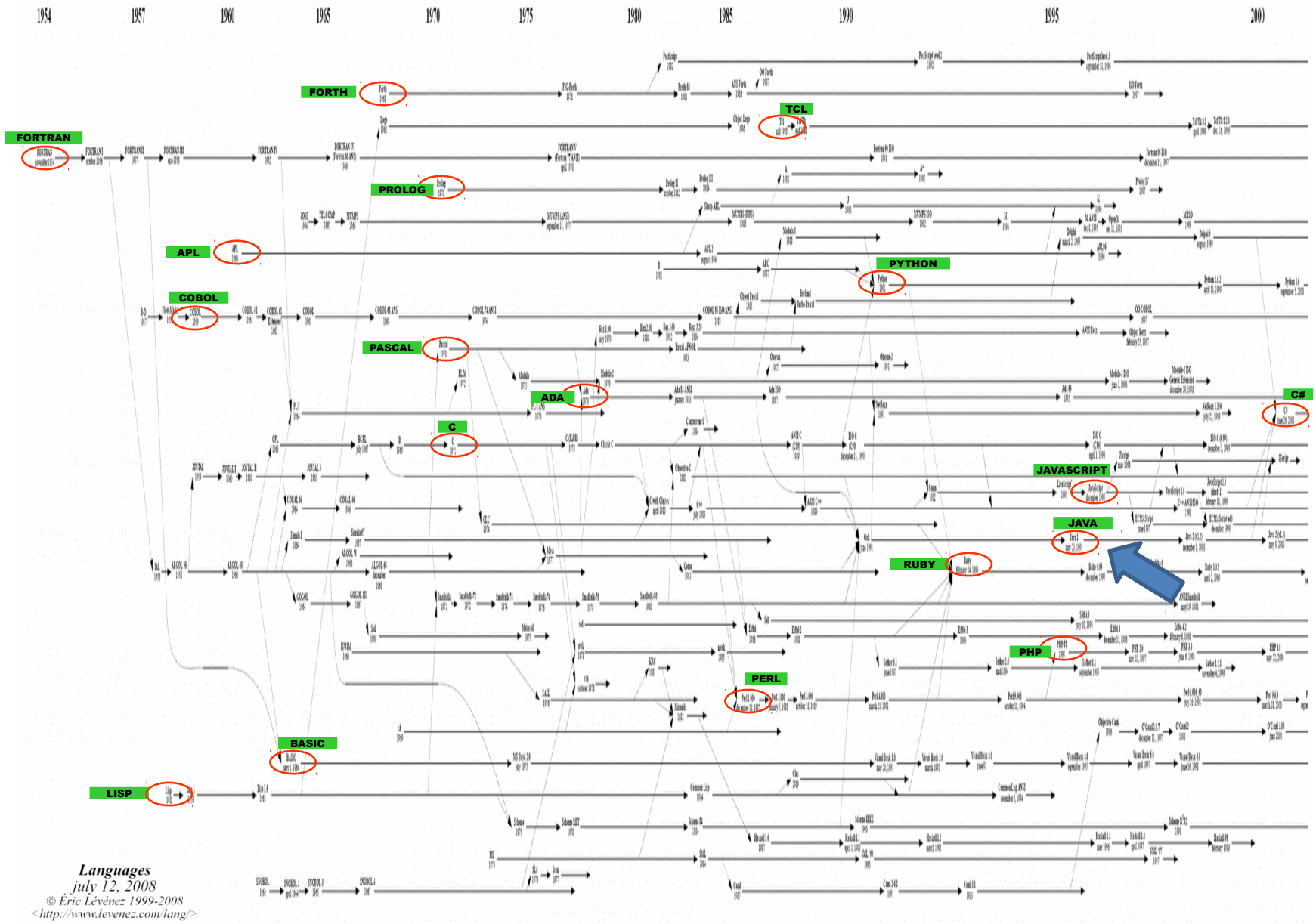
Libros: Computing machinery and intelligence, ¿Puede pensar una máquina?

Padres: Julius Mathison Turing, Ethel Sara Stoney

Hermanos: John Turing




A
B
Fortran
Algol 60
BASIC
CPL*
BCPL
Algol 68
Pascal
Simula 67
C
Mesa
Modula-2
Smalltalk-80
C++
CSP*
Ada
Pascal Concurrente
Distributed Processes
Lisp
ISWIM*
Prolog
Scheme
ML
Common Lisp
CLOS
Standard ML
sh, csh,ksh,..
REXX
PERL
TCL
Python
Oberon
Visual BASIC
JavaScript
"C++ ++ --"
JAVA
2014 – Java 8
60
70
80
90



http://www.levenez.com/lang/

¿ES JAVA LENTO?

Es ya una cuestión muy antigua y resuelta, pero sirve para ilustrar aspectos de la ingeniería del software: programar no es sólo escribir programas

¿Java para cómputo intensivo?

Esta página contiene apreciaciones discutibles.
(pero las conclusiones son VERDAD)

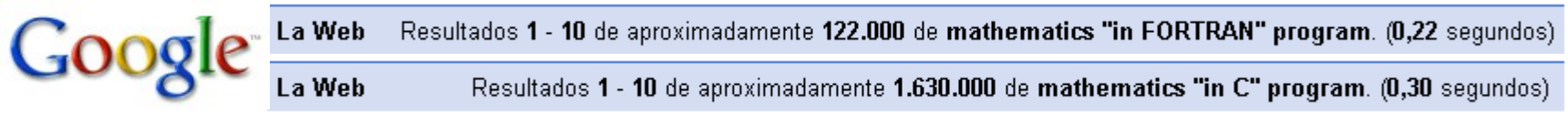
(X=matemáticas, física, ingeniería…)
Tradicionalmente, en computación para X, se ha buscado la "velocidad".

Esta visión X=cómputo intensivo puede ser cierta pero quizás parcialmente.

En todo caso MUCHÍSIMAS de las necesidades de computación "al límite" de hace unos años, hoy en día son livianas o "razonables". (mi conjetura: El mundo de lo "intratable" por "impotencia computacional" se ha reducido enormemente)

Esta ¿obsesión?/¿necesidad? justificaba el inmovilismo de las X frente a nuevos lenguajes (debería llevarles a programar directamente los microprocesadores en su lenguaje ensamblador, pero curiosamente no se daba esto).

Resultado: FORTRAN es la referencia, y el razonable paso a C ¿se dio?.



(comparación inválida sin duda. La presencia "arrasadora" de C junto a la "inteligencia" de Google (que utiliza el sinónimo "math" en la búsqueda) potencian el segundo resultado.

A Java se le ha "acusado" desde un principio de ser LENTO.

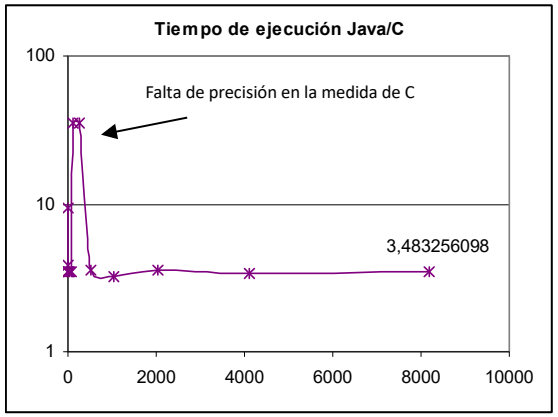
- Al principio era cierto.
  - Relación 4/1 frente a C
  - Razón principal: lenguaje interpretado
  - Otras razones: recogida de basuras, mecanismos de seguridad, etc.
- Desde hace unos años es comparable a C, dependiendo de en qué tareas. Ciertamente no es el mejor caso el del cómputo intensivo (estimaciones de un estudio particular 2004).
  - En gráficos bate a C
  - Relación media: 1.4/1 frente a C si excluimos gráficos
  - En cálculo intensivo la diferencia es más acusada
- Actualmente Java es más rápido que C en muchas tareas (particularmente gráficos) y similar en los peores casos (cómputo intensivo)
  - Máquina HotSpot
- Java (los lenguajes sobre máquinas virtuales en general) serán los más rápidos en el futuro (ante cómputos complejos, no para el caso de algoritmos muy "cerrados")
  - El futuro de las máquinas virtuales

cómputo intensivo?... Una experiencia concreta

Ejemplo peor caso Java vs. C (14ago08)
(cálculo de PI por MonteCarlo)

Experimento a partir del código tomado de http://husnusensoy.blogspot.com/2006/06/c-vs-java-in-number-crunching.html

- Comparación del tiempo de ejecución



La relación de tiempo de ejecución es del orden de 3,5 a favor de C

- Comparación del tiempo de preparación del experimento

JAVA:
- copiar, pegar, compilar, ejecutar y **listo en unos segundos**.

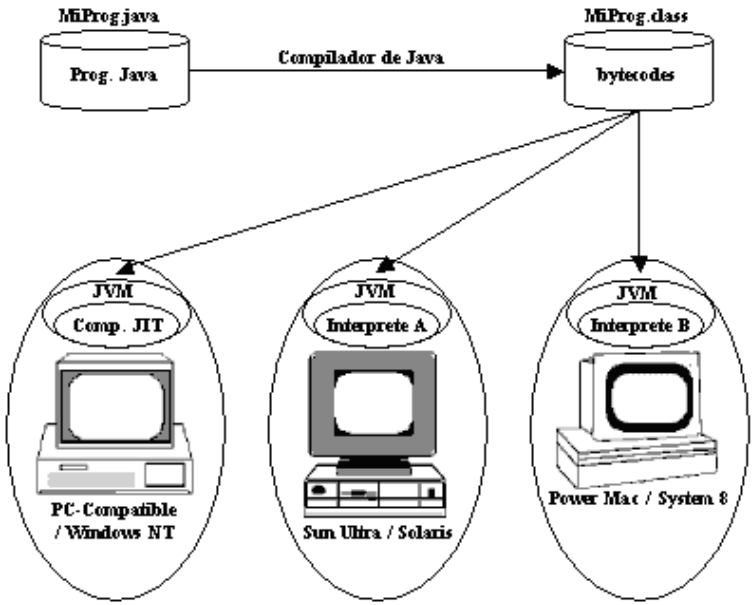
C:
- copiar, pegar, compilar, errores… (no coincide exactamente el lenguaje)
- corregir fuente, compilar, ejecutar, errores… (la arquitectura de la máquina no es la adecuada)
- corregir fuente, compilar, ejecutar, se observar falta de resolución de la función “time”,
- ir a la bibliografía para resolver el tema, no encontrar solución...
- replantear con iteraciones para obtener tiempos mayores…
- cambiar fuente compilar, ejecutar… errores de apuntadores (falta de práctica de un “ex” de C)
- corregir fuente, compilar, ejecutar… errores de violación de segmentos
- corregir fuente, compilar, ejecutar y… **listo en una hora**.

La relación de tiempo de preparación ha sido de 120 a favor de Java

LA MÁQUINA VIRTUAL

En la máquina Virtual está el "secreto"...
y quién quiera puede innovar.

"WRITE ONCE, RUN ANYWHERE"



Proprietary/closed source implementations
* Hewlett-Packard's Java for HP-UX, OpenVMS, Tru64 and Reliant (Tandem) UNIX platforms
* J9 VM from IBM, for AIX, Linux, MVS, OS/400, Pocket PC, z/OS
* Mac OS Runtime for Java (MRJ) from Apple Inc.
* JRockit from BEA Systems acquired by Oracle Corporation
* Oracle JVM (also known as "JServer" and as "OJVM") from Oracle Corporation
* Microsoft Java Virtual Machine (MS JVM) from Microsoft
* PERC from Aonix is a real time Java for embedded
* JBed from Esmertec is an embedded Java with multimedia capabilities
* JBlend from Aplix is a Java ME implementation
* Excelsior JET (with AOT compiler)

Lesser-known proprietary JVMs
* Blackdown Java (port of Sun JVM)
* CVM
* Gemstone Gemfire JVM - modified for J2EE features
* Golden Code Development (EComStation and OS/2 port of Java RTE and SDK for J2SE v1.4.1_07)
* Tao Group's intent
* Novell, Inc.
* NSIcom CrE-ME
* HP ChaiVM and MicrochaiVM
* MicroJvm from Industrial Software Technology (running of wide rande of microcontrollers 8/16/32-bit)

Free/open source implementations

* AegisVM	* JamVM	* Juice	* Mika VM
* Apache Harmony	•Jaos * JC	* Jupiter JVM	* Mysaifu JVM
* CACAO	* Jikes RVM	* JX (operating system)	* NanoVM
* IcedTea	* JNode	* Kaffe	* SableVM
* IKVM.NET	* JOP	* leJOS	* SuperWaba
* Jamiga			* TinyVM

* JESSICA (Java-Enabled Single-System-Image Computing Architecture)
* Squawk virtual machine (Sun JVM for embedded system and small devices)
* Sun Microsystems' HotSpot
* VMkit of Low Level Virtual Machine
* Wonka VM
* Xam

WIKIPEDIA
The Free Encyclopedia

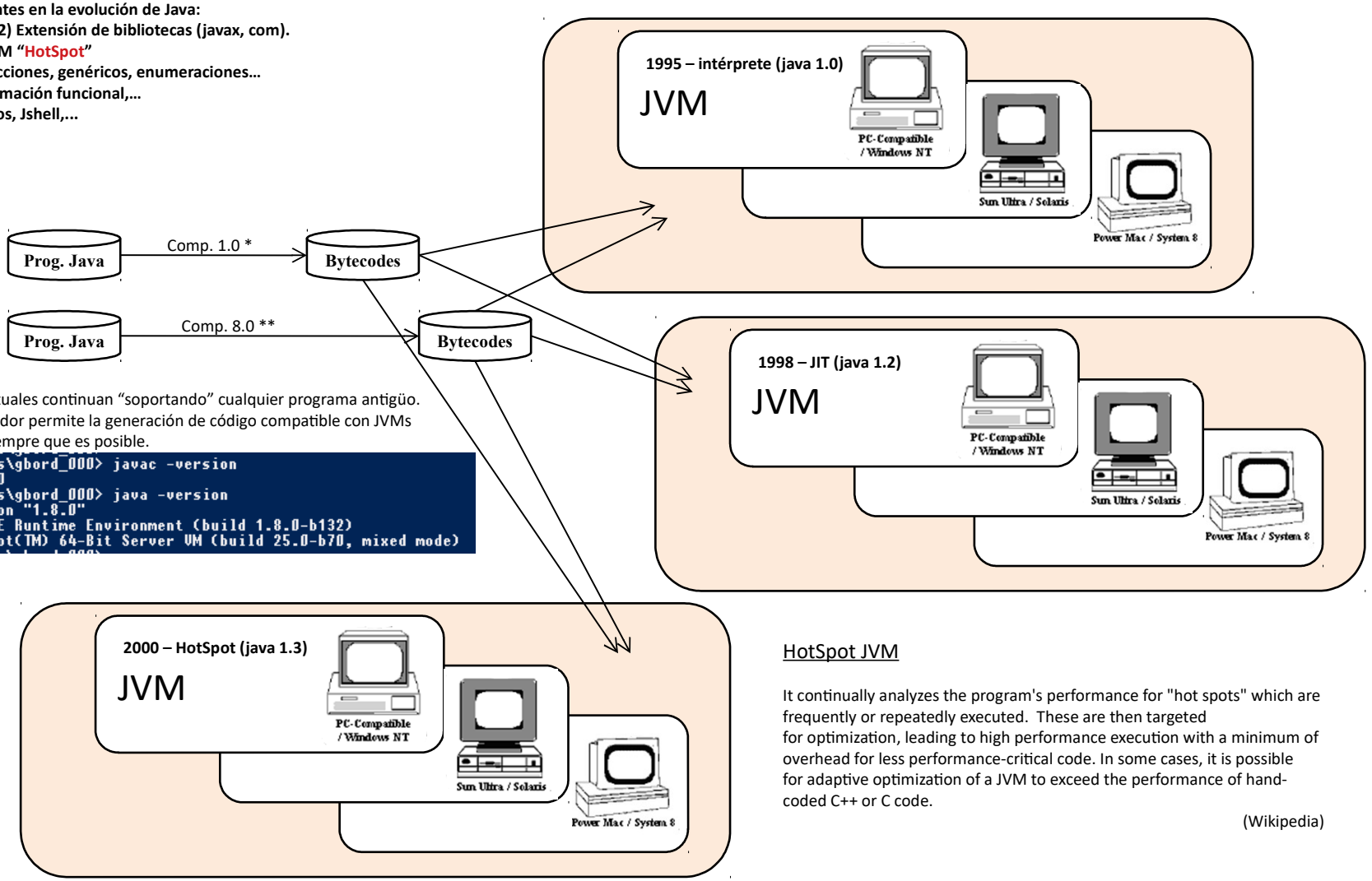
- Una idea novedosa, pero no del todo: cierta similitud con los lenguajes con código intermedio.
- Sí fue novedoso el enfoque de emulador de máquina (y la compilación JIT).
- Ventajas:
  - se pueden incluir con facilidad técnicas que en un diseño hardware pueden resultar prohibitivas por su complejidad técnica,
  - la posibilidad de evolución es mucho más sencilla al no requerir cambios de hardware
  - permite utilizar las "plataformas" existentes sin implicar una ruptura con los sistemas actuales (existe la máquina real pero…).
- **el diseño es público y la "implementación" es privada** (**especificaciones técnicas que debe cumplir toda JVM.**).
  - Distintos comportamientos en términos de velocidad y uso de memoria

Comentario 21/9/18:
La que han liado
"los de Python"

En la máquina Virtual está el "secreto"...
además aporta otras muchas ventajas.

Cambios importantes en la evolución de Java:

- **Java 1.2 (Java2) Extensión de bibliotecas (javax, com).**
- **Java 1.3 La JVM "HotSpot"**
- **Java 1.5: colecciones, genéricos, enumeraciones…**
- **Java 8 Programación funcional,…**
- **Java 9 Módulos, Jshell,…**



* Las JVM actuales continuan "soportando" cualquier programa antigüo.
** El compilador permite la generación de código compatible con JVMs anteriores siempre que es posible.

```
PS C:\Users\gbord_000> javac -version
javac 1.8.0
PS C:\Users\gbord_000> java -version
java version "1.8.0"
Java(TM) SE Runtime Environment (build 1.8.0-b132)
Java HotSpot(TM) 64-Bit Server VM (build 25.0-b70, mixed mode)
```

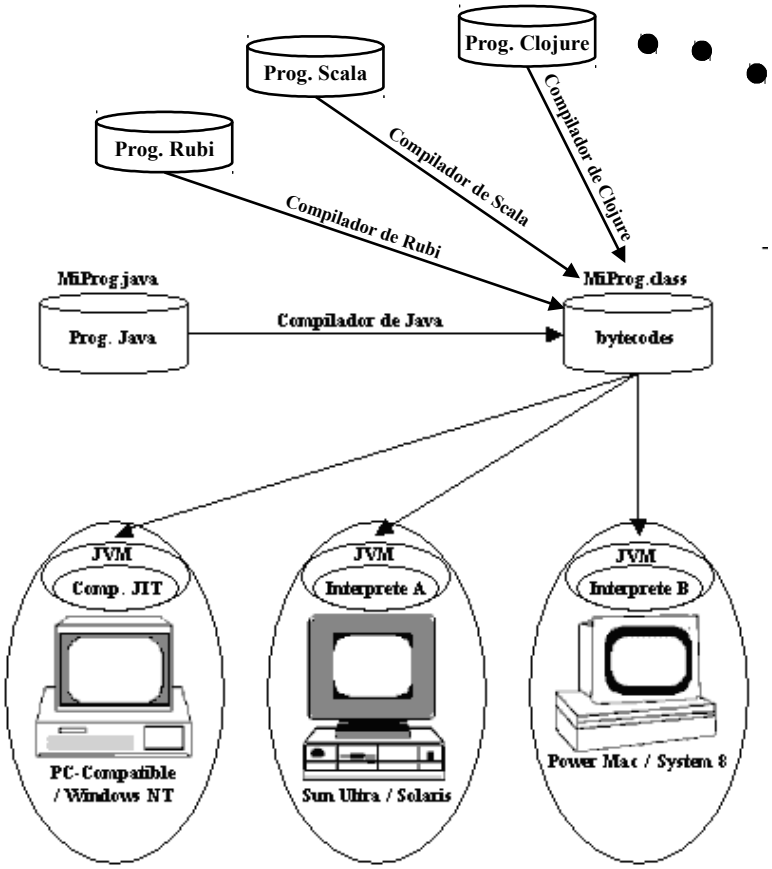
HotSpot JVM

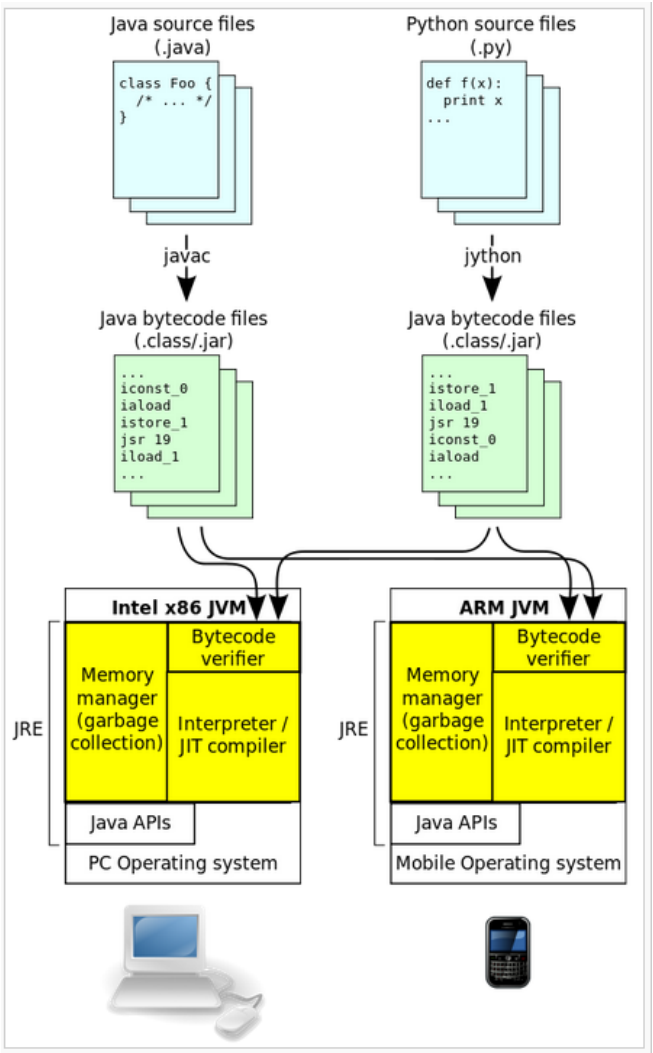
It continually analyzes the program's performance for "hot spots" which are frequently or repeatedly executed. These are then targeted for optimization, leading to high performance execution with a minimum of overhead for less performance-critical code. In some cases, it is possible for adaptive optimization of a JVM to exceed the performance of hand-coded C++ or C code.

(Wikipedia)

En la máquina Virtual está el "secreto"...
y no debemos confundirla con el lenguaje.

Comentario 21/9/18:
En JavaMagazine
hablan de la GraalVM





Wikipedia: Java virtual machine (sep 2014)

Versions of non-JVM languages

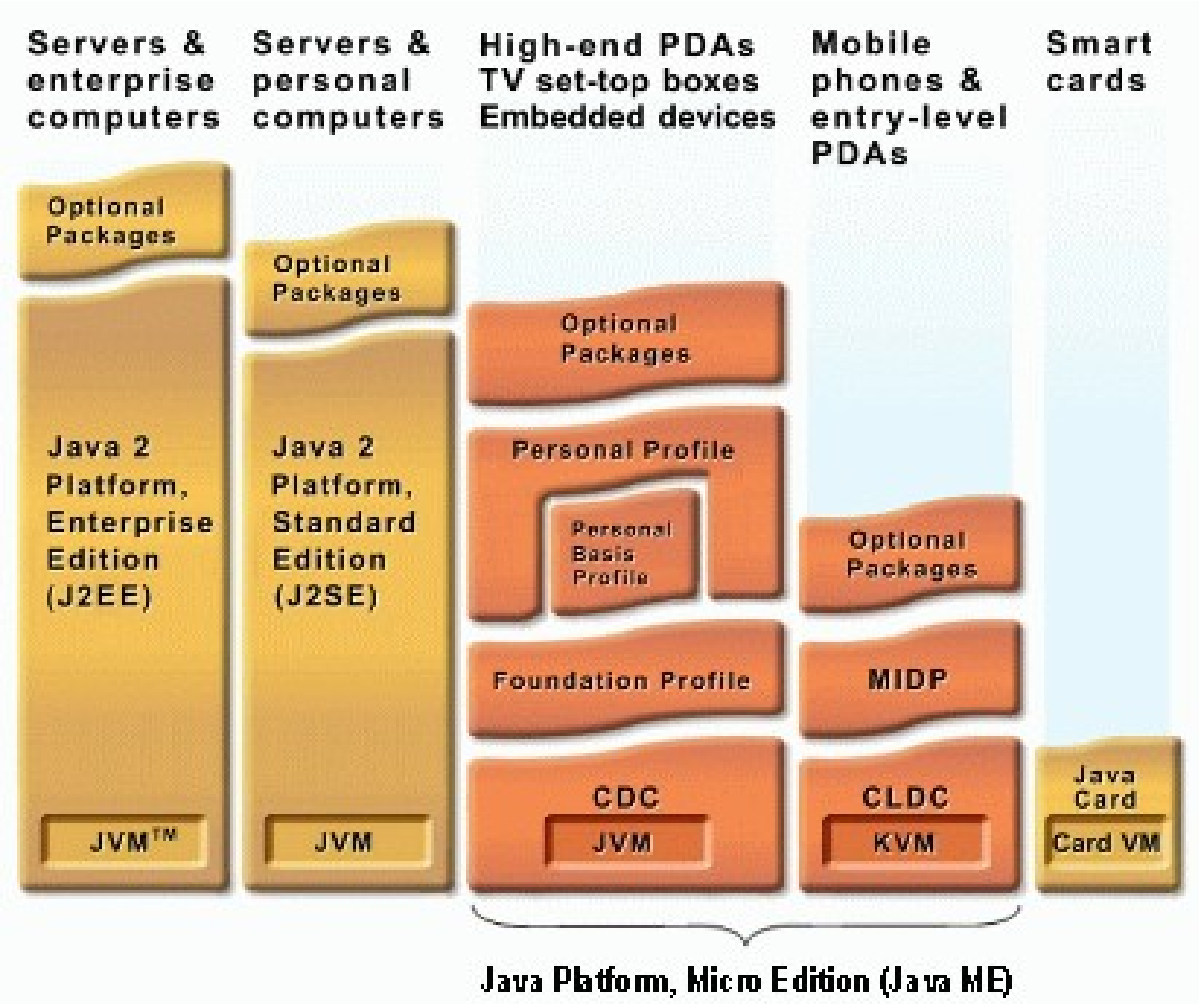
Language	On JVM
Erlang	Erjang
JavaScript	Rhino
Pascal	Free Pascal
PHP	Quercus
Python	Jython
REXX	NetRexx[3]
Ruby	JRuby
Tcl	Jacl

Languages designed expressly for JVM

Language
BBj
Clojure
Fantom
Groovy
MIDletPascal
Scala
Kawa

Wikipedia: Java virtual machine (sep 2014)

Hay (al menos) tres "grados" de Máquinas Virtuales Java



...luego esta la de Android (Dalvik)

(ahora ART)