

# Mono: la nueva plataforma de desarrollo Open Source

---

¿Qué es Mono?

Pablo Orduña  
-aka NcTrun-  
[pablo@ordunya.com](mailto:pablo@ordunya.com)

Abril 2005 – DotNetGroup  
*ESIDE – Universidad de Deusto*

This work is licensed under the Creative Commons Attribution-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305,





# ¿Qué es Mono?

- Mono es una **plataforma libre de desarrollo multiplataforma**
- Desarrollo dirigido por Novell/Ximian
- **Objetivo:** que GNU/Linux triunfe en el escritorio
  - Intentando dar a los desarrolladores de software de GNU/Linux una plataforma en la que desarrollen más software en menos tiempo



# Un poco de Historia - I

- Ximian

- Fundada por Miguel de Icaza y Nat Friedman
- Su propósito era desarrollar software de escritorio para GNU/Linux





# Historia - II

- Según afirman, esto les suponía gran esfuerzo
  - Crearon herramientas para facilitarlo
    - Sin obtener el resultado esperado
- Ya anteriormente habían llegado a la conclusión de que no se estaba reutilizando código en GNU/Linux
  - Cada lenguaje reimplementa sus librerías desde cero
  - Cada programa tenía poco más que libc o xlib en común con el resto

# Historia - III

---

- Evolution

- Aplicación grande
- Desarrollo demasiado caro
  - 2,5 años
  - 17 programadores en algunos momentos
    - de Icaza “los últimos 6 meses fueron dolorosísimos porque encontramos todo tipo de problemas con haberlo hecho en C”



# Historia - IV

- Desde Gnome se proponían soluciones:
  - Bindings de sus librerías
    - Cada vez que se cambia algo hay que volver a hacer demasiado trabajo
    - Lenguajes menos populares sufren
  - Bonobo
    - Solución a creación de componentes reusables
      - Utilizando CORBA
      - Basado en interfaces establecidos
      - Independiente del lenguaje
    - Los resultados no fueron los esperados



# Historia - V

---

- Por entonces (2000), Microsoft publicaba la “.NET initiative”
  - .NET Framework
    - Una nueva plataforma de desarrollo
    - Nueva infraestructura para desarrollo de Servicios Web
    - Nuevas herramientas para la plataforma de desarrollo
  - Hailstorm, el sistema de single-signon centralizado de Passport
    - No implementado por Mono
    - DotGNU sí implementa una alternativa



# Historia - VI

- El .NET Framework
  - Resolvía los problemas que habían tratado de resolver, de una forma más organizada
  - Ofrecía:
    - Garbage Collector, Threading...
    - Un nuevo lenguaje de alto nivel, C#
    - Una potente librería
    - Soporte para múltiples lenguajes
    - Especificaciones ya publicadas

# Historia - VII

---

- Las partes más complejas estaban estandarizadas
  - Permitiendo que se hiciesen implementaciones de ello
- A mediados de 2001 decidieron lanzar el proyecto Mono
  - Desarrollando una implementación libre del .NET Framework



# Características Básicas - I

---

- La plataforma es independiente del lenguaje
- Cuenta con un lenguaje universal, el CIL
  - Common Intermediate Language, también llamado IL o MSIL
  - Fácilmente compilable
  - Cada lenguaje tiene su compilador que genera CIL

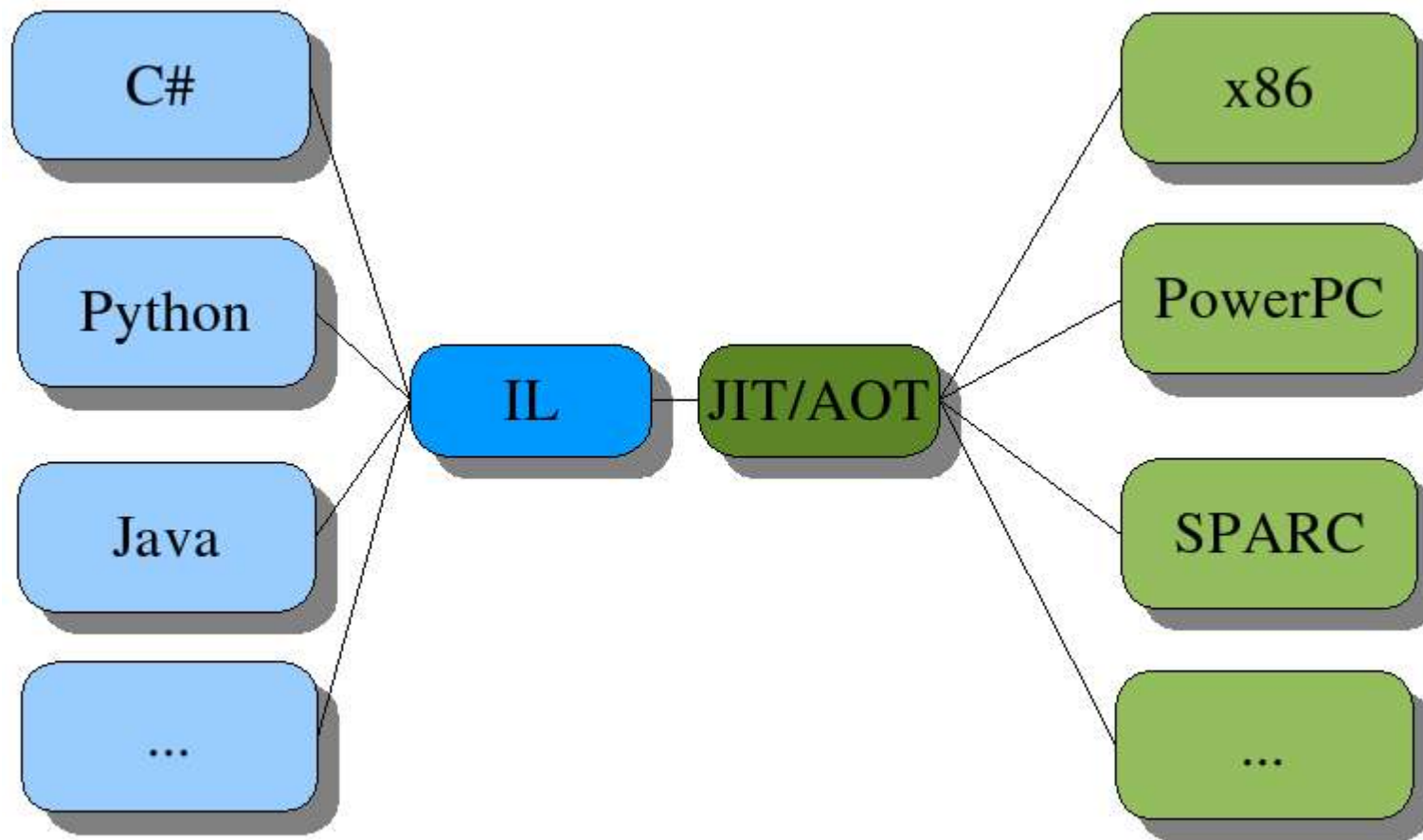
# Características Básicas - II

---

- Luego, del CIL se genera el código nativo de la plataforma en la que se ejecute
  - compilador JIT (Just In Time) o AOT (Ahead Of Time) o intérprete.
  - En estos momentos, están soportadas:
    - x86 (Linux, Windows, BSD, Solaris)
    - PowerPC (Mac OS X 10.2, 10.3, Linux)
    - AMD64 (Linux)
    - SPARC, S390



# Características Básicas - III





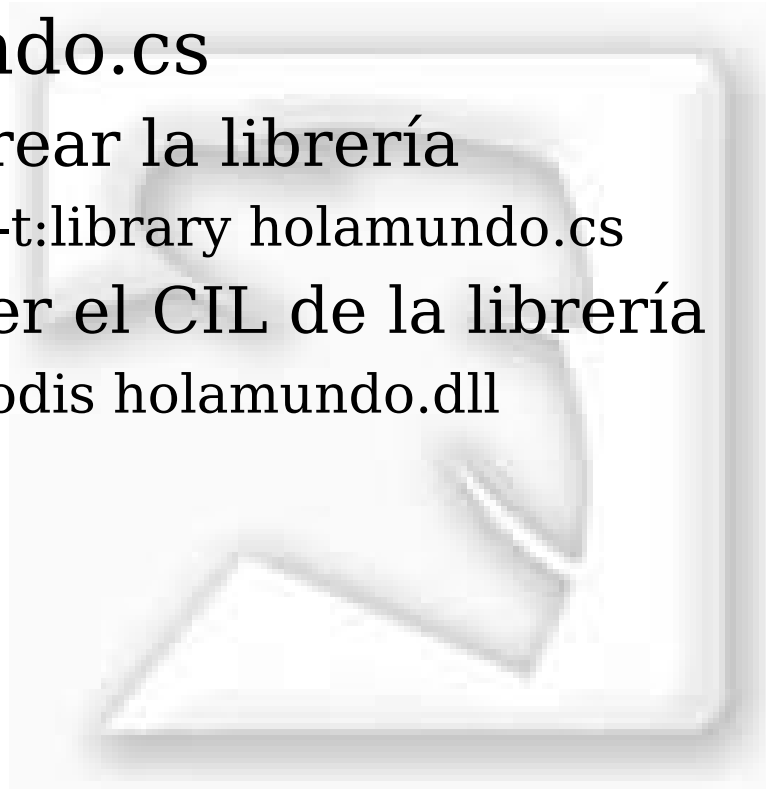
# Ejemplos - I

- Los ejemplos en el directorio “quees”
- Un hola mundo (dir. “holamundo”):
  - holamundo.cs
    - Para compilarlo:
      - mcs holamundo.cs
    - Para ejecutarlo:
      - mono holamundo.exe
    - Para ver el CIL:
      - monodis holamundo.exe
    - Para generar un shared object:
      - mono --aot holamundo.exe
    - Para ver todos los pasos intermedios del JIT:
      - mono -v -v -v -v holamundo.exe



# Ejemplos - II

- Generación de librerías (dir “librerías”):
  - holamundo.cs
    - Para crear la librería
      - `mcs -t:library holamundo.cs`
    - Para ver el CIL de la librería
      - `monodis holamundo.dll`





# Ejemplos - III

- Utilizando librerías (dir “utilizando”):
  - Utilizando holamundo.dll
    - Desde C#
      - `mcs -r holamundo DesdeCS.cs`
    - Desde Nemerle
      - `ncc -r holamundo DesdeNemerle.n -o DesdeN.exe`
        - `cs2n DesdeCS.cs DesdeNemerle.n` genera el `.n`
    - Desde VB.NET (alpha)
      - `mbas -r holamundo DesdeVB.vb`
        - Me da error, funciona si pongo:
          - `mbas -r $PWD/holamundo.dll DesdeVB.vb`



# Ejemplos - IV

---

- Desde IronPython
  - `ironpython DesdeIronPython.py`
  - `export MONO_PATH=$MONO_PATH:(dir bin de ip)`
  - `mono __main__.exe`
- Desde Boo
  - Compilando:
    - `booc DesdeBoo.boo`
    - `export MONO_PATH=$MONO_PATH:(dir bin de Boo)`
    - `mono DesdeBoo.exe`
  - Dinámicamente:
    - `booi DesdeBoo.boo`



# Ventajas

- Con esto:
  - Se puede desde un lenguaje utilizar componentes escritos en otro lenguaje
  - Dada una librería, su API es accesible a todos los lenguajes
  - Lo único que es necesario es que estos lenguajes tengan su compilador a CIL
  - Incluso se puede reutilizar los compiladores de terceros (IKVM, Boo...)
- ¿Y cuál es la API?



# Mono y .NET - I

- Mono se basa en el .NET Framework
  - De la misma forma que la FSF en su día inició el proyecto GNU basándose en UNIX
- Implementa varias APIs:
  - La de .NET
  - Extensiones de Mono



# Mono y .NET - II

- Microsoft estandarizó parte del .NET Framework
  - En los estándares ECMA 334 (C#) y ECMA 335 (CLI – Common Language Infrastructure)
    - Estandarizó el núcleo del .NET Framework y C#
  - C# ahora está estandarizado por la ISO
- Pero no todo está estandarizado
  - ADO.NET
  - Windows.Forms
  - ASP.NET
  -



# Mono y .NET - III

---

- Mono implementa casi todo el .NET Framework
  - También lo no estandarizado (por compatibilidad)
    - ADO.NET, ASP.NET...
  - Algunas partes no implementadas
    - Para algunas no van a dedicar esfuerzos (pero se acepta código de terceros)
      - System.EnterpriseServices o System.Management, por ejemplo
    - Otras están ya prácticamente implementadas
      - System.Windows.Forms por ejemplo



# Mono y .NET - IV

- Por tanto, son plataformas compatibles en muchos aspectos
  - Lo compilado en una plataforma puede ser ejecutado en la otra, siempre y cuando:
    - Sean aplicaciones 100% .NET: no utilicen P/Invoke
    - Utilicen librerías que estén portadas a la otra plataforma
    - Utilicen versiones compatibles
    - Cumpla lo básico en portabilidad ('/' o '\\'...)



# Mono y .NET - V

- Esta compatibilidad tiene grandes ventajas:
  - Reutilizar inversiones de Microsoft en documentación
    - Libros, artículos, cursos de .NET
    - Páginas web, foros, etc. dedicadas a .NET
  - Facilita la migración de proyectos desarrollados en .NET a GNU/Linux
    - Empresas que usan .NET y se plantean migrar a GNU/Linux
    - Utilizar componentes que otros programaron para .NET



# Librerías Mono - I

- Además, Mono desarrolla otras librerías, algunas de las cuales son:
  - Cairo
  - Bindings de Gnome (GTK#, Glade#...)
  - Bindings de aplicaciones
    - Evolution#, Mozilla, OpenOffice, iFolder...
  - Posix
  - Librerías de Novell
  - BD
    - Además de OleDB, MS SQL y Oracle, proporciona para IBM DB2, MySQL, Postgress, Sybase, Tds/Tdscliente y SQLite

# Librerías Mono - II

- Y puede utilizar muchas otras librerías implementadas por otros, entre ellas:
  - Remoting.CORBA
  - #ziplib (archivos zip y tar)
  - GIGen (OpenGL)
  - SDL.NET
  - QT#
  - un largo etcétera

# ¿Qué trae Mono, entonces?

---

- **Compiladores**
  - C#, VB.NET (alpha), JScript (desarrollo)
- **CLR (Common Language Runtime)**
  - Tanto JIT como AOT
- **Varias APIs**
  - La de .NET
  - Bindings de Gnome, Linux y demás



# Estado Actual - I

- Sistema de versiones como GNOME
  - X.Y.Z
    - “Y” par = estable, “Y” impar = en desarrollo
- Versión en desarrollo actual: 1.1.6
  - Nuevos componentes que no había en la 1.0
    - Compilador de VB.NET
    - Windows Forms 1.1
    - Debugger
    - Generics
  - Algunos componentes de Microsoft Whidbey
    - C# 2, .NET 2, XML 2, ASP.NET 2, ADO.NET 2, soporte para Puerto serie
  - Mejoras en rendimiento

# Estado Actual - II

---

- Próximas versiones:
  - 1.2
    - Mediados de 2005
    - Versión estable de lo que ahora es la rama 1.1
  - 2.0
    - Año 2006

