51CTO.com 技术成就梦想 | UBM

2012 / 10 / 25 ► 2012 / 10 / 26
中国 · 北京
October 25-26,2012
Beijing · China

World of Tech WOT

2012云计算架构师峰会

Cloud Computing Architects Summit China 2012

揭示企业级IT架构转型 分享最新技术的应用落地

主办单位：51CTO.com 技术成就梦想　联合主办：UBM

技术趋势和个人发展

陈皓

2012年10月

个人简介

- 行业背景
  - 金融行业(Thomson Reuters)
  - 计算平台(Platform)
  - 电子商务(Amazon)
- 技术背景
  - C/C++/Java
  - Unix/Linux/Windows
  - Web

Weibo: @左耳朵耗子
Twitter: @haoel
Blog: http://coolshell.cn/

我的个性

- 码农兼包工头
- 敏捷恐怖分子
- Unix/Linux/C/C++脑残粉
- C2C痛恨者
- CSDN腾讯百度批评人
- “技术部门无技术种族”歧视者
- 程序员文化民族主义者

大纲

- 编程语言的变迁
- 系统架构的变迁
- 技术人员的发展

如何了解技术发展的趋势

- 回顾历史，切勿追新
  - 朝着球的运动的方向去，而不是球的当前位置
- 注重基础，了解原理
  - 基础上的东西的变化少，基础上的东西一通百通
- 多看多想，多多实践
  - 国外的站点：Wikipedia, Hacker News, StackOverflow, GitHub, Reddit, Stanford Online Course, ….

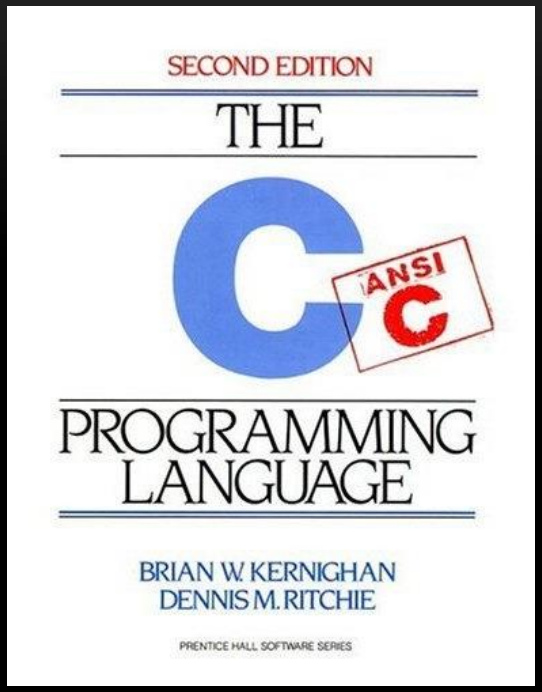
编程语言的变迁

主流语言的进化

- 静态语言
  - C → C++ → Java / C#
- 脚本语言（动态语言）
  - Shell（grep, sed, awk …）→ Perl / PHP / Ruby / Python
- 跨平台
  - 编译器→虚拟机 JVM→ 解释器→基于JVM的语言
- 编程方式
  - 面向过程→ 泛型 / 面向对象 / 函数式

C语言

- C语言可以学到什么?
  - 内存管理的基础
  - 程序编译的过程（预编译，编译，链接）
  - 程序的执行效率
  - 用C语言实现数据结构和算法
  - 操作系统的系统调用
- 学好C语言有什么用?
  - 很多语言都借鉴于C语言，如：C++,C#, D, Go, Java, JavaScript, Limbo, LPC, Objective-C, Perl, PHP, Python, Unix Shell
  - 了解系统底层，系统调优，任何东西都会反馈到操作系统层。



C → C++

- C++ 填C的坑
  - 结构体的内存问题（拷贝构造，赋值函数）
  - 宏的问题（const/inline/template）
  - 指针的问题（引用，RTTI）
  - 类型转换问题（四种cast）
  - 封装和重载问题
  - 资源回收问题（RAII – 智能指针）
  - 大量的if-else多种逻辑混在一起的问题（面向对象，泛型）
- C++ 的强大之处是“泛型编程”
- C++ 的危险之处是“滥用”

C/C++ → JAVA

- Java解决C/C++的问题
  - 指针的各种问题（引用）
  - 内存管理的各种问题（垃圾回收）
  - 错误处理（异常）
  - 纯面向对象（接口编程）
  - 跨平台问题（JVM）
  - 程序模块的耦合（反射,动态代理→IOC/AOP）
- Java的强大之处在于“面向对象”和“J2EE系统框架”
- Java的问题在屏蔽底层细节

Where the Focus Is

	Efficiency BOT – 1999, 2009 – EOT	Flexibility	Abstraction (OO, Generics)	Productivity (Automatic Services, Tools)
C = efficient high-level portable code. Structs, functions.	●	●	*non-goal*	*non-goal*
C++ = C + efficient abstraction. Classes, templates.	●	●	●	*non-goal*
Java, C# = productivity. Mandatory metadata & GC, JIT compilation.	*at the expense of*	*at the expense of*	●	●

动态语言

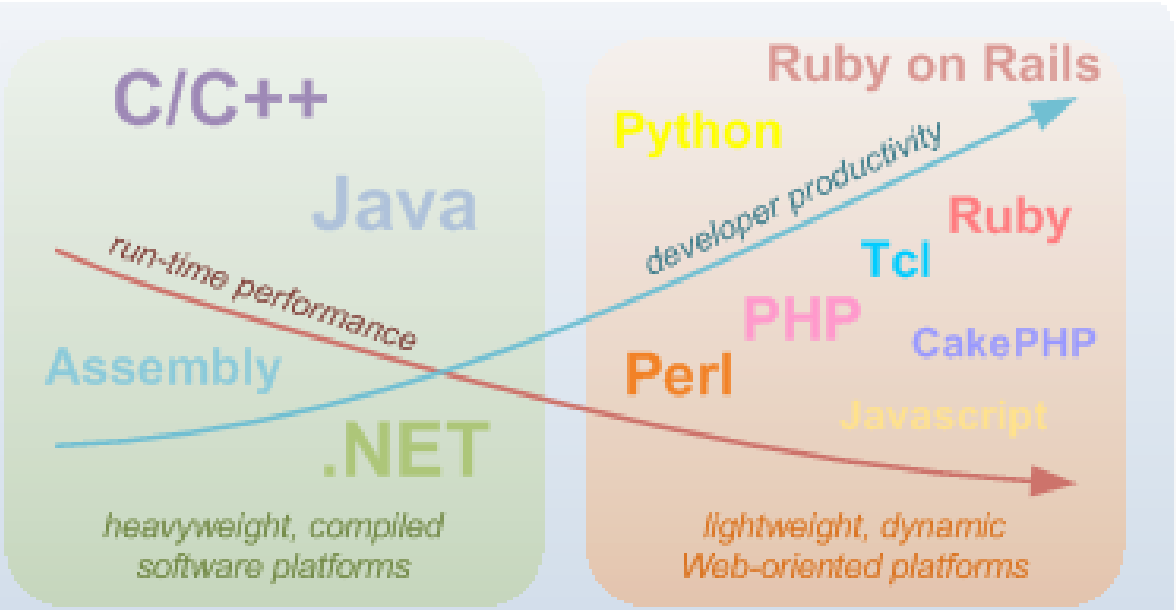
- **Python**
  - Mutable type
  - 无需链接，无需编译，模块拿来就用
  - 语言更简洁，数据操作更自然
  - 支持命令式编程，面向对象，函数式，面向切面，泛型
  - 完美地结合C, C++, Java 和Unix Shell
  - “用一种方法，最好只有一种方法来干一件事”

- **优势：生产率，自然，灵活……**
- **劣势：性能**

基于JVM的语言

- 企图使用JVM对脚本语言优化
  - 更好的GC，更好的异步I/O, JVM优化，JIT
- 主流的基于JVM的语言
  - 动态脚本：Jython, JRuby, Groovy
  - 静态语言： Scala

21st Century Web Development:
The classic struggle between developer productivity and run-time efficiency



Note: *"Platforms"* here means the language plus any run-time libraries/frameworks.

Source: Dion Hinchcliffe. http://web2.socialcomputingmagazine.com

系统架构的变迁

单机时代

- 数据库、SQL、业务逻辑、界面全在一台机器
- 一些技术
  - Foxbase / Foxpro
  - VB + Access
  - Delphi + Interbase

客户端/服务器时代

- 服务器端 → 数据库
- 客户端 → 界面，业务逻辑，SQL
- 主流相关技术
  - Powerbuilder + IDBC/ADO + SQL Server
  - Delphi + IDBC/ADO +SQL Server
  - C/C++ + EC + RDBMS

浏览器/服务器时代

- 服务器→UI, 业务逻辑，SQL，RDMBS
- 客户端→浏览器
- 主流相关技术
  - LAMP
  - IIS+ Delphi + CGI/ISAPI + ODBC/ADO + RDBMS
  - Tomcat+ JSP/Servlet + JDBC + RDBMS
  - IIS+ ASP + ODBC/ADO + RDBMS
  - ActiveX

三层结构

- 数据库服务器 → 数据存储
- 应用服务器 → 业务逻辑，SQL
- Web服务器 → UI
- 主流相关技术
  - J2EE - Websphere / WebLogic
  - 中间件 IBM CICS, BEA Tuxedo
  - RPC
  - COM, CORBA

分布式计算

- 数据库服务器分布式
- 应用服务器分布式
- Web前端服务器分布式
- 相关主流技术
  - 数据库同步、分区。
  - 缓冲机制。NoSQL – MongoDB, Redis …
  - 消息机制。JMS, MessgeQueue, Thrift …
  - 异步机制。Workflow Engine, Pub/Sub …
  - 负载均衡。
  - 分布式一致性。
  - P2P技术。

云

其它技术

其它技术

- 操作系统 – POSIX 标准
- 网络协议 – TCP/UDP – Socket
- I/O模型（异步）
- 设计
  - 模块依赖 → 接口依赖
  - 低耦合，高内聚，拼装
- 测试/部署（自动化）
- 数据库（RDBMS → NoSQL → RDBMS）
- 前端（PC → Web → 移动 → Web）

怎么面对技术

技术方面

- 语言（逻辑控制）
- 算法+数据结构（数据处理）
- 系统（内存，文件，I/O，网络，进程/线程，UI）
- 设计（代码组织，模块组织）
- 工具（开发调试，版本管理，测试，部署，监控）

软件开发的“三重门”

- 1、业务功能–粗放地开垦（劳动密集型公司）
  - 使用各种编程语言工具堆功能
- 2、业务性能–扩大化生产（技术型公司，工程师文化）
  - 深入了解技术的原理和基础
- 3、业务智能–精耕细作（创新型公司）
  - 机器学习，数据挖掘，算法，数据，统计学，人工智能.....

态度方面

- 技术无贵贱，不要挑食
  - 前端和后端一样，都是编程。前端侧重用户的嗅觉，后端侧重各种机制原理的深入。
- 小心“我会在我需要的时候再学”
  - 你不可能学习那些你以为不存在的东西
  - 是人都能做网站，但不是每个人都能做出支持百万用户的网站
- 广度的知识是深度研究的副产品（wikipedia）
  - 死记硬背 vs 深度研究
- 和高手工作
  - 重要的是你要让高手想和你一起工作

态度方面

- 对技术有热情就是不给自己找借口
  - 我没有时间，我太忙，所以我没学
  - 我没有经历过这样的项目，所以我不会
  - 对于某些事情，如果以前没有在你身上发生过，那么这个事情在未来也不会发生。

- 挑战无处不在
  - 那怕是一个很小的功能做到极致都有很大的挑战
  - 我们的身边有很多很多的东西都应该让我们去思考去求解

选择

- 户口，薪资，相比起你的人生经历，你的眼界，你的发展，什么都不是。
- 眼界和经历最重要。价值并不仅仅是名利权。
- 和有激情能做事的人做有意义的事。

三个问题

- 早晨，是什么驱动着你开始新的一天?
- 你现在的正在经历的有没有让你感到兴奋?
- 你经历过的有没有让你觉得没有荒度?

谢 谢！