程序开发:目前主流开发技术的分析和总结 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/245/2021_2022__E7_A8_8B_E5_BA_8F_E5_BC_80_E5_c97_245952.htm 主流的程序设计语言：C 、Delphi(ObjectPascal)、Java、C# 桌面应用程序框架：MFC、VCL、QT、JavaAWTSWING、.Net 企业应用程序框架：WindowsDNA（ASP、COM、COM ）、J2EE 、.NetFramework 开发工具：VisualBasic、Delphi、VisualC 、C Builder、VisualC# *程序设计语言：C Delphi（本来应该是ObjectPascal，但为了简单，我就语言和工具混为一谈吧）JavaC#(虽然他刚刚推出，但因为微软为之倾注了大量心血，一定会成为一种重要的开发语言) *桌面应用程序框架:MFCVCL *企业应用程序框架:WindowsDNAJ2EE.Net *COM技术：我单独提出这项技术，是因为它无法简单的被视为语言、桌面应用程序框架或企业应用程序框架，它与这些都有关系。 2.1 程序设计语言 2.1.1 C 语言的演进 最初要从二进制代码和汇编说起，但那太遥远了。我们就从面向过程的语言说起吧（包括BasicCFortranPascal）。这种面向过程的高级语言终于把计算机带入了寻常的应用领域。其中的C语言因为它的简单和灵活造就了Unix和Windows这样的伟大的软件。 面向对象的语言是计算机语言的一个合乎逻辑的进化，因为在没有过多的影响效率、简单性的前提下提供了一种更好的组织数据的方法，可使程序更容易理解，更容易管理这一点可能会引出不同意见，但事实胜于雄辩，C 终于让C语言的领地越来越小，当今还活着的计算机语言或多或少的都具备面向对象的特征，所以这一点并不会引起太多困惑。C

的成功很大程度要归因于C，C 成为它今天的样子是合乎逻辑的产物。因为在面向过程的时代，C几乎已经统一天下了。今天著名的语言象JavaC#都从C借鉴了很多东西，C#本来的意思就是C 。其实C 曾经很有理由统一面向对象程序设计语言的天下来着，但可惜的是，C 太复杂了。即使是一个熟练的程序员，要你很清楚的解释一些问题你也会很头痛。举几个还不是那么复杂的例子来说： 对=的重载成员转换函数拷贝构造函数转化构造函数之间有什么区别和联系呢？ 定义一个类成员函数private:virtualvoidMemFun()=0.是什么意义呢？ int(*(*x(int))[4])(double).是什么意思？ 还有其他的特征，比如说可以用来制造一种新语言的typedef和宏（虽然宏不是C 的一部分，但它与C 的关系实在太密切了），让你一不小心就摔跤的内存问题（只要new和0delete就可以了吗？有没有考虑一个对象存放在容器中的情况？）……诸如此类，C 是如此的复杂以至于要学会它就需要很长的时间，而且你会发现即使你用C 已经好几年了，你还会发现经常有新东西可学。你想解决一个应用领域的问题比如说从数据库里面查询数据、更改数据那样的问题，可是你却需要首先为C 头痛一阵子才可以，是的，你精通C ，你可以很容易的回答我的问题，可是你有没有想过你付出了多大的代价呢？我不是想过分的谴责C ，我本人喜欢C ，我甚至建议一个认真的开发普通的应用系统的程序员也去学习一下C ，C 中的一些特性，比如说指针运算模板STL几乎让人爱不释手，宏可以用几个字符代替很多代码，对系统级的程序员来说，C 的地位是不可替代的，Java的虚拟机就是C 写的。C 还将继续存在而且有旺盛的生命力。 2.1.2 Java和C# Java和C#相对于C 的不同最大的有两

点：第一点是他们运行在一个虚拟环境之中，第二点是语法简单。对于开发人员而言，在语法和语言机制的角度可以把Java和C#视为同一种语言。C#更多的是个政治的产物而不是技术产物。如果不是Sun为难微软的话，我想微软不会费尽心力推出一个和Java差不多的C，记得Visual J 吗，记得WFC吗？看看那些东西就会知道微软为Java曾经倾注了多少心血。而且从更广泛的角度来说，两者也是非常相似的C#和Java面对的是同样的问题，面向应用领域的问题：事务处理、远程访问、Webservice、Web页面发布、图形界面。那么在这一段中，我暂且用Java这个名字指代Java和C#两种语言尽管两者在细节上确实有区别。Java是适合解决应用领域的问题的语言。最大的原因Java对于使用者来说非常简单。想想你学会并且能够使用Java需要多长时间，学会并且能够使用C 要多长时间。由于Java很大程度上屏蔽了内存管理问题，而且没有那么多为了微小的性能提升定义的特殊的内容（比如说，在Java里面没有virtual这个关键字,Java也不允许你直接在栈上创建对象，Java明确的区分bool和整型变量），他让你尽量一致的方式操作所有的东西，除了基本数据类型，所有的东西都是对象，你必须通过引用来操 作他们；除了这些之外，Java还提供了丰富的类库帮助你解决应用问题因为它是面向应用的语言，它为你提供了多线程标准、JDBC标准、GUI标准，而这些标准在C 中是不存在的，因为C 并不是直接面向解决应用问题的用户，有人试图在C 中加入这些内容，但并不成功，因为C 本身太复杂了，用这种复杂的语言来实现这种复杂的应用程序框架本身就是一件艰难的事情，稍后我们会提到这种尝试COM技术。渐渐的，人们不会再用C 开发应用领域的软

件，象MFCQTCOM这一类的东西最终也将退出历史舞台。 2.1.3 Delphi Delphi是从用C 开发应用系统转向用Java开发应用系统的一个中间产物。它比C 简单，简单的几乎象Java一样，因为它的简单，定义和使用丰富的类库成为可能，而且Delphi也这么做了，结果就是VCL和其他的组件库。而另一方面，它又比运行于虚拟环境的Java效率要高一些，这样在简单性和效率的平衡之中，Delphi找到了自己的生存空间。而且预计在未来的一段时间之内，这个生存空间将仍然是存在的。可以明显的看出，微软放弃了这个领域，他专注于两方面：系统语言C 和未来的Java(其实是.Net)。也许这对于Borland来说，是一件很幸运的事情。如果我能够给Borland提一些建议的话，那就是不要把Delphi弄得越来越复杂，如果那样，就是把自己的用户赶到了C 或Java的领地。在虚拟机没有最终占领所有的应用程序开发领域之前，Delphi和Delphi的用户仍然会生存得很好。 2.2桌面应用程序框架 目前真正成功的桌面应用程序框架只有两个，一个是MFC，一个是VCL，还有一些其他的，但事实上并未进入应用领域。遗憾的是我对两个桌面应用程序框架都不精通。但这不妨碍我对他做出正确的评价。 2.2.1MFC MFC（还有曾经的OWL）是SDK编程的正常演化的结果，就象是C 是C的演化结果一样。MFC本身是一件了不起但不那么成功的作品，而且它过时了。这就是我的结论。MFC凝聚了很多天才的智慧当然，OWL和VCL也一样，侯捷的《深入浅出MFC》把这些智慧摆在了我们的面前。但是这件东西用起来估计不会有人觉得很舒服，如果你一直在用Java、VB或者Delphi，再回过头来用MFC，不舒服的感觉会更加强烈。我不能够解释MFC为什么没有能

够最终发展成和VCL一样简单好用的桌面程序框架，也许是微软没精力或者没动力，总之MFC就是那个样子了，而且也不会再有发展，它已经被抛弃了。我有时候想，也许基于C这种复杂的语言开发MFC这样的东西本身就是错误的可以开发这样的一个框架，但不应当要求使用它的人熟悉了整个框架之后才能够使用这个系统，但很显然，如果你不了解MFC的内部机制，是不太可能把它用好的，我不能解释清楚为什么会出现这种现象。 2.2.2VCL 相比之下VCL要成功的得多。我相信很多使用VCL的人可能没有像MFC的用户研究MFC那样费劲的研究过VCL的内部机制。但这不妨碍他们开发出好用好看的应用程序，这就足够了，还有什么好说的呢？VCL给你提供了一种简单一致的机制，让你可以写出复杂的应用程序。在李维的Borland故事那篇文章中曾经说过，在Borland C 3.1推出之后Borland就有人提出开发类似C Builder一类的软件，后来竟未成行。是啊，如果C Builder是在那个时候出现的，今天的软件开发领域将会是怎么样的世界呢？真的不能想象。也许再过一段时间，这些都将不再重要。因为新生的语言如Java和C#都提供了类似于VCL的桌面应用程序框架。那个时候，加上Java和C#本身的简单性，如果他们的速度有足够块，连Delphi这种语言也要消失了，还有什么好争论的呢？只是对于今天的桌面程序开发人员来说，VCL确实是最好的选择。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com