Borland 传奇拾遗

关于 Borland 公司最后时光的非官方记忆

Lex Li

Borland 传奇拾遗

关于 Borland 公司最后时光的非官方记忆

Lex Li

This book is for sale at http://leanpub.com/borlandlegendmore

This version was published on 2017-03-18

Contents

第一章黑暗中的灯火 1

第一章黑暗中的灯火

在 JBuilder 登上王座，Delphi 5 复兴成功之后，Borland 的一段艰难时光似乎过去。但新世纪的 IT 技术日新月异，所为决策者应该怎样去洞察并开拓新的领域来推动公司下一步成长是一项十分紧迫的任务。那么 Borland 公司接下来发布了哪些新产品来应对挑战？这些新产品又是否得到了市场的追捧呢？

Kylix 的横空出世与快速衰落

什么是 Kylix？Kylix 乃是在 Dale Fuller 在执政 Borland 初期一手打造的 Linux 平台 RAD（Rapid Application Development）工具。《Borland 传奇》一书中于第三章和第四章已经提及了这个项目的策划和开始阶段。Borland 开发 Kylix 这个新产品，意图就是跳出 Windows 平台的束缚，借由打入 Linux 平台来开拓新领地。

图 **1 Kylix** 盒装版

值得注意的是 JBuilder 虽然可以藉由 JVM（Java virtual machine）运行在 Linux 上面，但因为使用它开发的 Java 程序仍需要 JVM 才能在 Linux 平台运行，所以严格意义说它并非原生开发工具。Kylix 作为 Borland 公司第一款完全针对 Linux 平台原生程序开发的作品，在下面几个方面独树一帜：

- Delphi 和 C++ 的编译器能在 Linux 平台上运行，能够将 Object Pascal 和 C++ 源代码编译成适合 Linux 平台的原生机器码。
- Kylix 是一个十分类似 Delphi 的集成开发环境，能够进行快速程序开发。
- Kylix 开发的应用基于 Delphi RTL 和全新的基于 Qt 的 CLX 框架，能够同时支持 Windows 和 Linux。

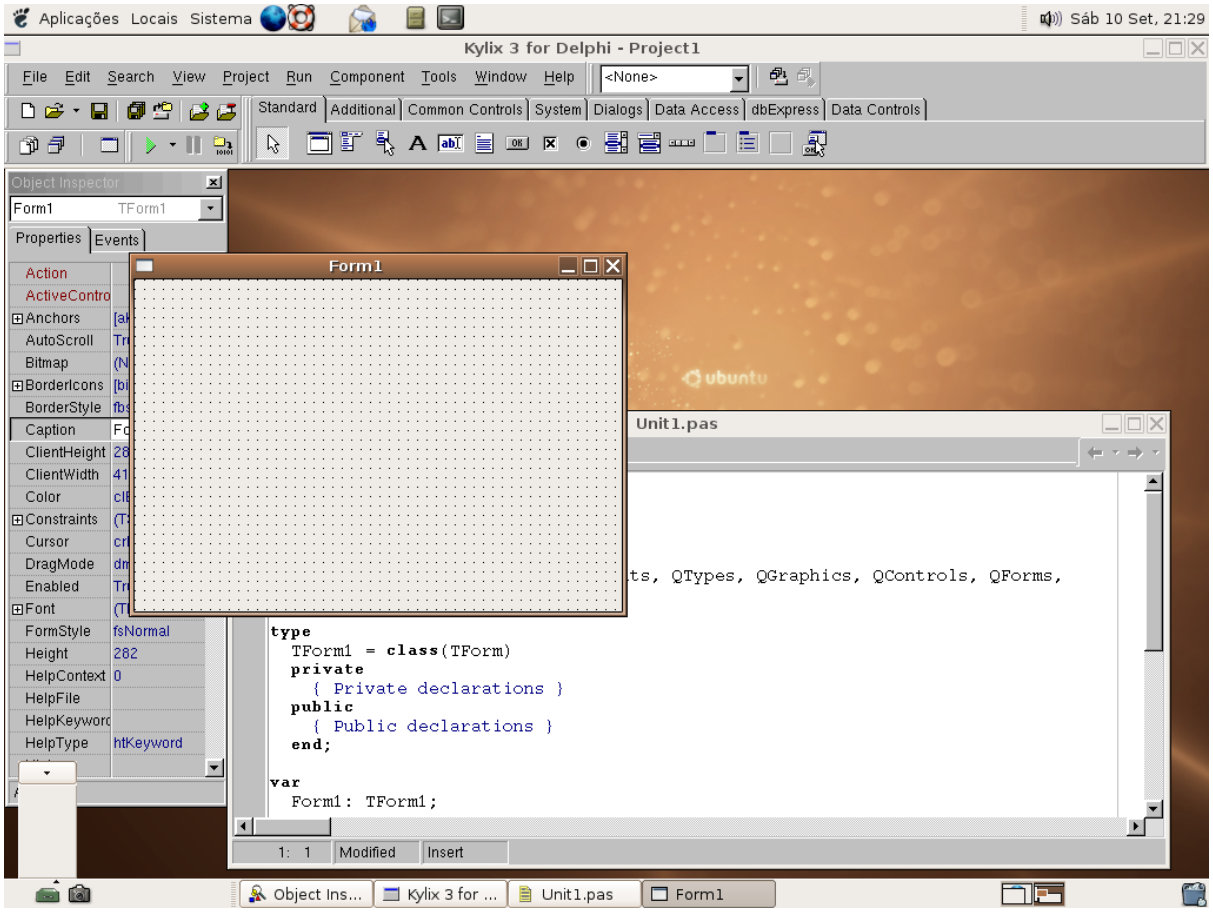

图 **2** 运行在 **Ubuntu** 上的 **Kylix**

参考 Delphi 的版本和售价，Borland 确定 Kylix Developer 版本售价 999 美元，而更高级的 Server 版本售价 1999 美元。另外 Kylix 还有一个只能用于 GPL 开源软件开发的 Open Edition，是可以从 Borland 官方网站免费下载的。Borland 官方也组织编写了一本 Kylix 开发指导书籍,

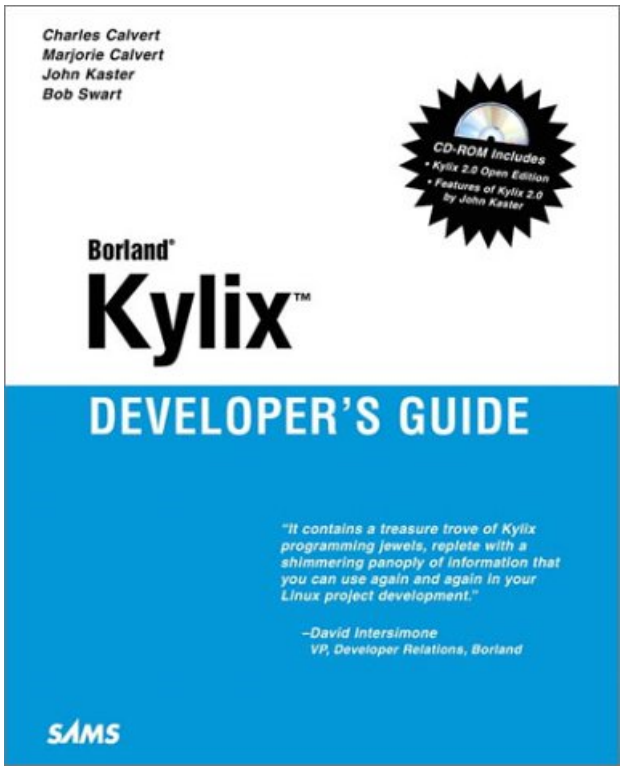

图 **3 Borland Kylix Developer's Guide** 封面

单从产品特性来看，Kylix 的确提供了很多 Windows 版本 Delphi 才有的快速开发支持，而且能够直接运行在 Linux 操作系统上，这在产品经理看来应该具有大卖的潜力。那么产品发布之后销量情况如何呢？从有限的资料来看，实际的销量不仅不如意，而且口碑情况也很差。这又是什么原因呢？

首先从产品自身设计来看，Kylix 开发中的多个重要抉择后来都被证明并不恰当：

- Kylix IDE 基本是稍经改造的 Delphi 5 IDE，其实是藉由 Wine 运行在 Linux 平台上。对于 Wine 的依赖带来了 IDE 运行缓慢和不稳定的问题——要知道直到多年之后的零八年六月十七日 Wine 项目才发布它的 1.0 版本。两千年左右基于 Wine 来做这么大型的商业级别应用，Borland 的研发团队实际上冒着多大的技术风险呀！

图 **4 Wine** 图标

- 由于 Kylix 编译出来的应用程序以及搭配的低阶调试器都是 Linux 原生应用而 IDE 部分却是运行在 Wine 上面，在进行应用程序集成调试时实际上整个过程都涉及到复杂的进程间通信。过大的复杂度使得这一部分代码一直问题多多，使用者不得不面对调试缓慢并且中途极易崩溃的问题。在软件开发过程中调试显然是一个十分重要的环节，Kylix 不解决调试方面的问题就无法达到用户的期望。

- 基于 Qt 的 CLX 新框架。CLX 全名为 Component Library for Cross Platform，发音类似 clicks（VCL 则念为 vicle）。VCL 设计之初虽然精巧，但是仅仅考虑了和 Windows API 配合，因此几乎不能迁移到其他操作系统平台。为了使 Kylix 程序开发能有类似 VCL 级别的开发框架，Borland 的工程师基于 Qt 库作为基础来从头设计了 CLX。可是 Kylix IDE 自己不使用 CLX 开发的，Borland 以外的公司也没有谁使用 CLX 做成了重量级的应用，因此这个框架的可用性一直受到质疑。大量的控件厂商也一直对 CLX 持观望态度，不敢贸然投入研发资源，因为优秀的控件包也几乎看不到。

图 **5 Qt** 图标

Qt 确实是设计十分精良的框架，但是从之后多年的评价来看，基于任何跨平台的界面框架（包括 SWT/Swing/Qt）的程序运行在各种操作系统上时，都达不到原生界面库程序那种和操作系统浑然一体的感觉。

想做到与平台的深度集成，应用程序开发者还是必须在每个系统上使用内建的界面开发框架。Mono 便是一例：Windows 上界面使用 WinForms，Linux 上使用 GTK#，Mac 上 MonoMac，iOS 上 MonoTouch，Android 上 Mono for Android。当然这些真知灼见都是多年之后才为众人认同。

话说 Qt 也是个命运多舛的产品。Qt 诞生之初就显示出优秀的显示效果，以至于在它还未基于 GPL 发布时 Linux 社区就以它为基础开发出 KDE 桌面系统。后来为了解决 KDE 对于这个不开源框架的依赖，又导致了 GTK 和 Gnome 桌面的诞生。虽然 Qt 后来也加入了 GPL 授权，Nokia 买下 Trolltech 之后进一步增加了 LGPL 授权，希望能够扩大 Qt 的受众范围，但是由于 Linux 桌面和 Symbian 操作系统都最终跌落成小众市场，这个框架也逐渐失去了主流地位。

> 零二年八月 Qt 再次易手。芬兰公司 Digia 从 Nokia 手中买下了 Qt。希望这家公司能够好好经营，不要像 Nokia 一样浪费了手里大把的好技术。

从外部原因来看，还有多个因素或多或少对这个产品造成了伤害：

- Borland 擅长的是封闭的商业开发工具模式。这种模式和 Linux 这个开源平台似乎从一开始就是场门不当户不对的姻缘。Linux 平台虽然缺少优秀的集成开发平台，但是不缺少的是各种免费的开发工具。这首先就影响了 Kylix 的受众面。Borland 也因此主要以 Delphi for Linux 为主题来向自己培育多年的 Delphi 社区推广，从时任 Borland Rad Tools Group 副主席和总经理的 Simon Thornhill 先生这封公开信便可见一斑，http://edn.embarcadero.com/article/22904。
- Linux 在千禧年前后红极一时，似乎只要和它沾边的股票或者公司都能行情看涨。这也是使得 Borland 下定决心赌一把 Kylix 的重要原因。但是这种热潮居然很快就伴随着股市泡沫破裂而退去，不仅使得 Kylix 这个新产品的前景变得十分不明朗，也进一步打压了销售。历史也证明在此后的多年里面尽管 Linux 多次试图在桌面市场对抗微软的 Windows 平台，它在这个领域仍然不是 Windows 的对手。而同期苹果公司的 Mac OS X 平台却凭借功能和品质上的稳步提升显著地提升着市场占有率。
- 主流的 Linux 新发布版本不断发布。习惯了 Windows 平台的开发者会有一个明显的体验，那就是在微软的控制下这个平台一直是十分稳定的周期发布新版本——而且从 Vista 开始这个周期有拉长的趋势。所有 Windows 软件一般可以有条不紊的进行升级换代。另外微软平台的兼容性也是有目共睹的优秀。可是 Linux 除去有着林林总总的分发版本，核心类库的升级也很频繁，甚至会出现明显的不兼容（例如 GTK 和 Qt 的大版本升级）。由于 Kylix 中对于 Linux 不少函数库都采用自己的 Patch，而这些 Patch 都没有提交 Linux 社区，或由于种种原因没有被接纳。因此 Borland 工程师必须自己花更多精力去更新和管理这些 Patch，追踪种种新问题，然后进行繁琐的测试。这一方面增加了维护成本和产品的风险，也直接影响到 Kylix 的稳定性和产品质量。

站在那个时代的高度来看 Kylix 的确是一个理念上很有特色的产品。但是它的惨淡销量和所带来的收益比之巨大的投入来说实在是令决策者失望。也就是靠着 Delphi 5/6/7 三个版本还不错的销售收入，Borland 才坚持 Kylix 产品的开发到零二年。因此在 Borland 最终决定 Delphi 要全面进入.NET 战场后，Kylix 的开发也也被叫停，最后版本停留在 3.0。

局外的 **CrossKylix** 和 **Free Pascal Compiler**

由于 Kylix IDE 很快就不能支持最新的 Linux 分发版本，Simon Kissel 设计并发布了能够在 Delphi 和 Windows 上交叉编译 Kylix 和 Linux 代码的 CrossKylix[1] 项目。

[1] http://crosskylix.untergrund.net/

图 **8 CrossKylix** 图标

使用这个插件，我们可以完全在 Windows 上面工作，而将最终交叉编译的产品部署到 Linux 上进行测试。这不仅绕过许多使用 Kylix 的问题，也大大简化 Kylix 开发的流程。但是 CrossKylix 缺少远程调试支持，所以无法调试程序是始终困扰用户的问题。只是此时 Borland 已经无暇升级 Kylix，或者给 Simon 提供帮助。由于 Kylix 的回归遥遥无期，这个项目在零四年十月宣布终结（有意思的是一零年七月二十一日这个项目又发布了 1.1.0 版本）。

伴随着 Borland Kylix 的沉寂，Free Pascal Compiler[2] 这个开源项目却在 Linux 平台开发这方面逐渐赶超过去:

- 相比起仅仅针对 Linux 平台的 Kylix，FPC 支持的操作系统平台更多，包含了 UNIX 和 Mac OS X，iOS 及 Android 等，甚至有点追赶 GCC 的味道了。
- FPC 配合跨平台的界面库 Lazarus Component Library (LCL)，最终开发出 Lazarus 这样一个跨平台的 IDE。

[2] http://en.wikipedia.org/wiki/Free_Pascal_Compiler

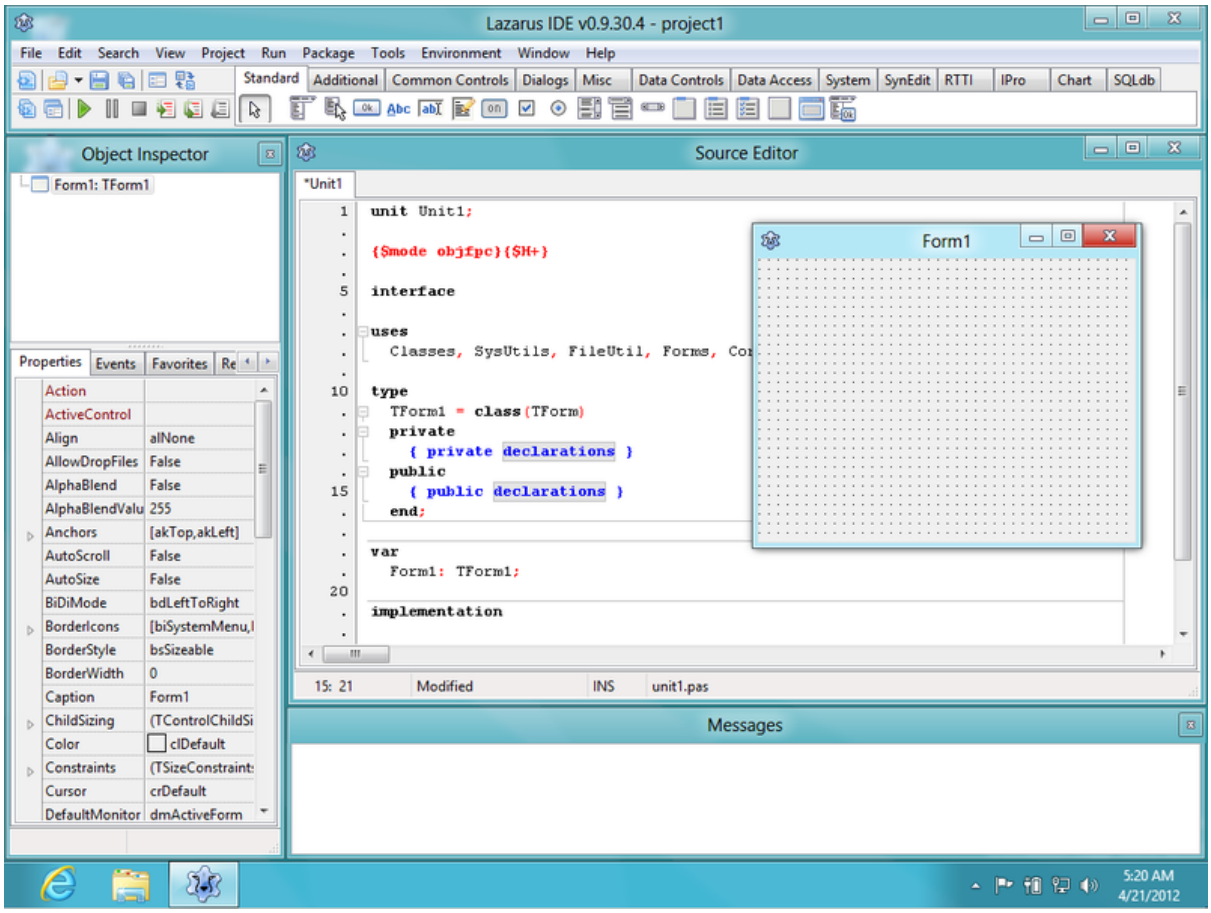

图 **10 Lazarus IDE**

Simon 曾宣布自己将尝试用 FPC（Free Pascal Compiler）取代 Kylix 来做一个叫做 CrossFPC[3] 的新项目。笔者十分期待将来能够见到它的发布。

图 **9 CrossFPC** 图标

或许最终 Delphi 会由于开发团队的解体和母公司的消失而消沉，但 FPC 一路伴随着 Turbo Pascal 和 Delphi 走来却是越变越强，应该会在未来 Pascal 语言领域扮演越发重要的角色。

Embarcadero 的逆袭

多年以后 Delphi 产品到了 Embarcadero 手中时，Mac OS X 已经在桌面市场取得了不错的占有率。Delphi 产品组也再次激活跨平台开发工作的计划：

- 开发环境完全是运行在 Windows 平台上的 Delphi IDE，而对于其他平台，提供远程调试功能。

[3] http://delphi.org/2010/07/43-simon-kissel/

- 跨平台开发框架摒弃了 VCL 和 CLX，而是采用了 FireMonkey。FireMonkey 类似微软的 WPF 框架，整个程序的 UI 部分完全自己绘制，只是模拟各个目标操作系统的主题。
- 将主要精力放在支持 Mac OS X，iOS 和 Android，而最后才开始慢慢支持 Linux。

这些改变多少都和 Kylix 的经验相关，也体现出产品经理 JT 对于未来市场竞争的思考。

C++BuilderX 的夭折

2003 年是 Borland 全力冲刺.NET 的一年，连续发布了 C#Builder，Janeva 以及 Delphi 8 for .NET 三个重要的产品——分别针对 C#，Java/.NET 互操作和 Delphi 市场。但是在 C++ 方面 Borland 公司却一直没有进一步的消息，这使得许多 Borland C++ 的用户都开始担心 C#Builder 是不是会取代 C++Builder 的地位。

一年后我们看到称为 X 一代的三个产品（JBuilder X，Delphi 2005 和 C++BuilderX）时，这种担心应该是稍有缓解，因为新的 C++ 开发工具又回来了。

图 **11 JBuilder X** 盒装版

可是这个新的产品却几乎成为了 Borland 历史上最有争议的一个作品，这又是为什么呢？

C++Builder 系列开发工具一直是 Delphi 产品线的附庸，以 C++ 语言配合 VCL 来进行快速程序开发。但是 VCL 这把双刃剑却时时刻刻制约着 C++Builder 在 C++ 方面的发展：

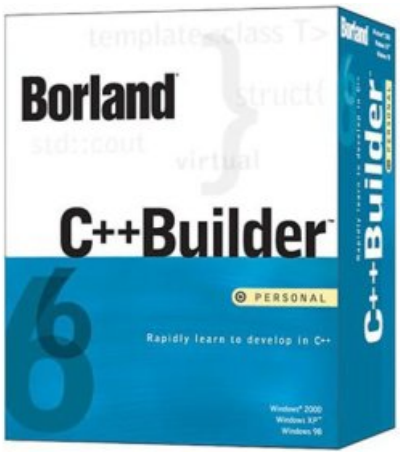

图 **12 C++Builder 6** 盒装版

- C++ 的跨平台支持十分好，各种操作系统平台都支持 C++ 语言开发。但是 C++Builder 仅仅支持 VCL 开发，因此被牢牢绑定在 Windows 平台。同时由于 VCL 仅有 32 位并且不支持 Unicode，C++Builder 也只能用来开发 32 位的应用程序，也不能支持 Unicode。
- C++ 有着标准库（STL）同时也有着众多的跨平台框架。但是 Borland 的 C++ 编译器发展很慢，对于 C++ 新标准（当时称为 C++ 0x，现在叫做 C++11）的支持一直不太完善。很多流行的框架如 Boost 也不能在 Borland 的 C++ 编译器下编译通过。

李维先生在《Borland 传奇》第十二章中预见了新版 C++ 开发工具的模样。笔者认为那是因为他已经提前了解到 Borland C++ 小组的新计划。在 C++BuilderX 面世的时候，可以看出这个新版 C++ IDE 十分符合李先生的描述。从 Borland 官方的文档来看，C++BuilderX 其实还有一个名字叫做 Borland C++ 2005，仿佛与辉煌的 Borland C++ 古老年代一脉相承。从此可见产品经理对这个新产品的定位之高。

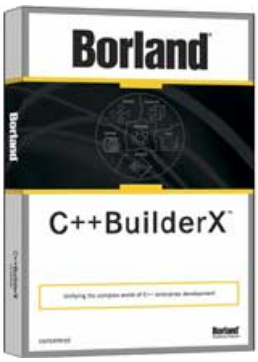

图 **13 C++BuilderX** 盒装版

这个新产品那时确实在业界引起了很大的关注，甚至是吸引了很多原本没用过 Borland C++Builder 的程序员。笔者寝室中一位多年从事 Java 开发的同学也第一时间下载了试用版。紧接着《程序员》杂志上面也刊登了侯捷先生的试用体验。为什么 C++BuilderX 能如此受到关注呢？下面是几个显而易见的优点：

- IDE 采用 JBuilder 的核心 PrimeTime，能够顺利运行于多个操作系统平台。较之 Kylix 仓促使用的 Delphi 5 IDE，PrimeTime 从稳定性到成熟度都要优秀很多。
- 支持多种 C++ 编译器。除了 Borland 自家的全新编译器，开发人员还可以选择使用 GCC 或者其他跨平台编译器来构建工程。
- 计划支持跨平台的 wxWindows 控件库（后改名为 wxWidgets），将能够进行跨平台快速程序开发。

纵观当时市面上 C++ 的 IDE，能够达到 CBX 这个高度的几乎没有：Visual C++ 肯定不能跨平台使用；Eclipse 的 CDT 有始终没有完善；Nokia 那时还没有买下 Qt 也就没有 Qt Creator。如此让人眼前一亮的新产品，为什么没有能够成为市场的执牛耳者并带动 Borland C++ 的复兴呢？多年之后审视这个产品，我们能够看到下面几个硬伤：

- 为了尽早上市，C++BuilderX 1.0 甚至没有一个正式窗体设计器。对于习惯了 Windows 平台上 RAD 开发工具的程序员来说，这肯定是一个致命问题。随盘附

赠的 Preview 版本虽然有窗体设计器原型，但是仅仅支持 Windows 平台和很少的几个标准控件，而且直到 CBX 1.5 这个问题产品组也没有解决。虽然如此粗暴的销售方式符合了 Borland 当时发布产品的不良习惯——总是把一些实际上并不那么成熟的产品当正式产品强行的拿出来卖——但是结果也使得那些曾经热切希望借 CBX 进行新项目运作的开发人员望而却步。即使后来的 BorCon 上面 C++ 小组表示让 CBX 支持 VCL 也不是不可能，但是想必这一过程不那么容易，不然后面为什么没有再接再厉发布 CBX 2？

- 帮助文档和资料稀缺。既然这个产品这么革命性，改了 IDE 又改了编译器支持，那么作为一个入门者必然需要充分的帮助文档来学习 IDE 使用，学习编译器配置。但是产品自带的帮助根本就没有详细到一看就会的程度，缺少太多基本的内容。笔者十分耐心的尝试学习使用 CBX 和调试，结果花了很多精力却没什么收获，不得不放弃。文档方面的问题后来也成为了 Borland 后续产品的通病之一，长期为人诟病。
- 更加重要的是 Borland 自己的 C++ 用户真正急需的是一个 C++Builder 6 的升级版来维护和改进现有项目。可是 CBX 1.0/1.5 既不支持 VCL/CLX，也不支持 BCB 的旧工程，完全不能满足老用户的需要。

公开信——Borland C++Builder 社区的呼声

不论 Borland C++ 产品经理有多么大的雄心壮志把 Borland C++ 带领到多少新平台上面，任何新产品计划都不可能以牺牲现有用户为代价。毕竟到了这个阶段，Borland 公司已经在 Windows 和 Java 两条主线上遇上了强大的竞争，一旦失去了 C++Builder 现有的用户，恐怕立即就会陷入万劫不复的境地。

这些 C++Builder 的用户也越来越感到 6.0 版本在网络时代程序开发过程中的种种局限，从而开始强烈要求 Borland 升级 C++Builder 以适应新潮流。这种声音在 BorCon 04 之后逐渐统一，并在零四年十月形成了一份公开信发表在 Borland 新闻组上。

图 **14 BorCon 04** 标志

Paul Gustavson 是这封信的执笔人。他同时也是 Borland C++Builder 社区中备受尊敬的作者、讲师和技术专家。

这封公开信当时被放置在 http://bcbjournal.org/community_letter.pdf 以接受用户们签名支持（由于时代久远，这个链接不再有效）。信中主要是列举了 CBX 的不如意之处，而且强烈要求 Borland 继续以 C++Builder 6 为基础开发新的 BCB。由于联名支持者众多，这样的一封公开信让 Borland 的管理层立即看到了问题之严重，也使得 Borland 于当年十二月中旬也正式以公开信的形式予以回应，http://edn.embarcadero.com/article/32845。

Borland 承诺将继续 C++Builder 的开发并将继续支持 VCL，初步打消了用户的担忧。这一变故最终推动了 C++Builder 2006 的推出，而之后接连推出的 C++Builder 新版本

也成功接棒 C++Builder 6。但它同时迫使 C++ 小组不得不停止了 CBX 开发，Borland C++ 当时的产品经理也挂冠而去。

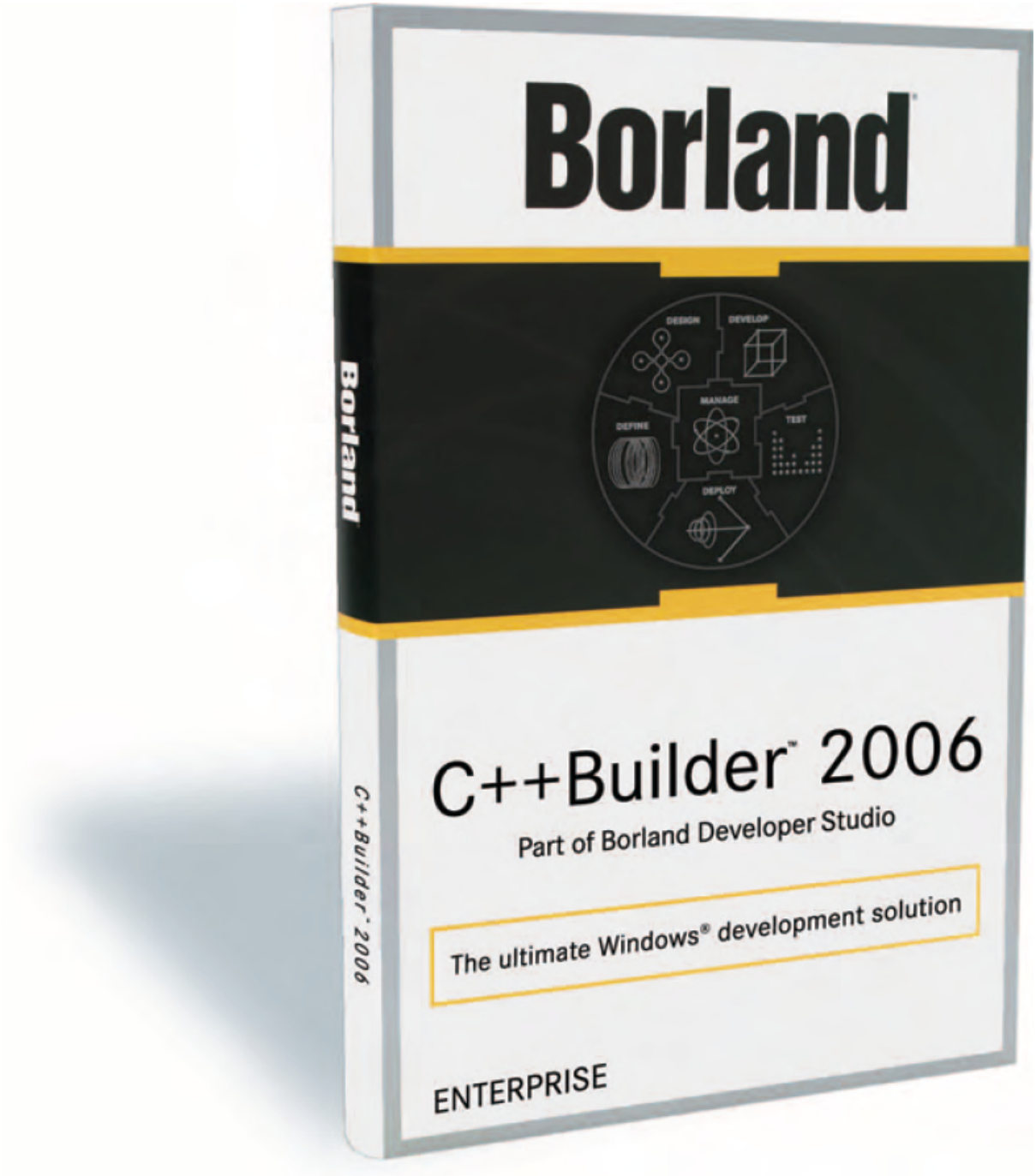

图 **15 C++Builder 2006** 盒装版

零五年十一月李维先生光临武汉来推介 Delphi 2006，还在学校念书的笔者专程跑去江北的香格里拉饭店参与其中。先生本人同书中照片一样可爱，只是脸上挂着些许的疲惫。虽然活动是宣传 Delphi 2006 新产品，不过业界更加关心的是 Borland 的前途问题。Dale Fuller 的下台，似乎进一步让 Borland 堪忧的运营状况都暴露在世人眼前。那时笔者尤其关注的还包括 Kylix 和 C++BuilderX 的命运。按李维先生当时给出的说法，Borland 内部正在考虑开源这两个项目。不过，从现在看来两个项目一个也没有开源，

多少让人感到些许遗憾。

作为一个力图摆脱 Delphi 阴影的新产品 CBX 至此也就该寿终正寝了；Delphi 2006 中也确实加入了 C++ 开发支持；Borland C++ 社区的怒火也些许的平息了。但是关于未来呢？C++ 发展的潮流究竟会如何呢？虽然众多的狂热 C++ 粉丝依然抗拒 Java 等其他语言平台，但是 Linux 上最方便的 C++ IDE 一度还是 NetBeans。因为这一类 Java 构建的 IDE 可以安装在众多的平台上，而且对于多种编译器和 Framework 的支持也是不错的，大大的减少了学习 IDE 使用的重复投资，而且使得很多 Java 开发的实践——例如重构，极限编程通过共用 IDE 的形式渗透到了 C++ 社区中。即使 Borland C++ 小组作出的努力没有能够帮助 CBX 取得胜利，但是相信日后东山再起也不是没有机会。一旦未来 CodeGear 重新投入足够的资源激活在 Eclipse CDT 平台上的新 C++ 开发工具，锻造出一个类似今天 JBuilder 2007 一样成功的 C++ IDE 来，Borland C++ 的复兴也就可以期待了。

> 当然以上是笔者年轻时候的想法了。最后做出这个跨平台 C++ 神器的毫无疑问是 JetBrains，它的 CLion 各个方面都做到了极致。现在连 Visual C++ 也支持 Linux 程序的开发和远程调试。只能说 C++ 的好时代又来了。

小结

在极短的时间内，Borland 的产品组连续推出了两个重量级的新产品力图拓展疆土。Kylix 和 CBX 的产品设计思路，都具有极高的前瞻性，而且打造的产品也的确体现了 Borland 研发部门的实力。但是 Borland 公司的大环境不佳，产品销售策略冒进，产品质量没法保证，目标市场缺乏活力和高成长，都最终导致了糟糕的销量和项目的搁浅。

另一方面，Delphi 向.NET 平台的转型一直不是非常成功，没有能够带来计划中大量的收入不说，老用户也拒绝升级。JBuilder 面对 Eclipse 的进攻也是只能维持现状，甚至开始节节败退。如果说当年 JBuilder 华丽的后发制人多亏了 Delphi 火爆销售的支撑，那么在 2000 年之后这两个主要产品自顾不暇，再没有办法给予新产品资源上的支持了。

既有的市场在快速失去，而新产品迟迟不能打开新的局面，Borland 也终于在 05-06 年度重新陷入了 Inprise 时代的类似困境中。不想这一次它终于没有再次走出绝境，而 Kylix 与 CBX 的惨淡结局为 Borland 公司的覆灭敲响了丧钟。