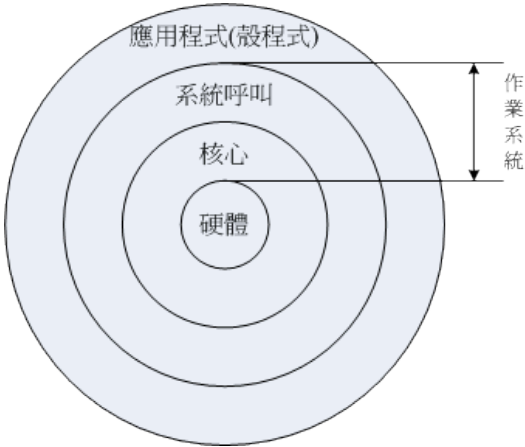
# Linux是什么？

## Linux是什么 操作系统/应用软件

操作系统处理**有效率的控制硬件资源的分配**，并**提供计算机运作所需要的功能**（如网络功能），还需要**提供程序设计师更容易开发软件的环境**（所以操作系统会提供一整组系统呼叫结构来给软件设计师开发使用）

Linux就是一套操作系统



由于 不同的硬件的功能函数并不一样 （例如ibm的power cpu与intel的x86就不一样 所以同一套操作系统无法再不同硬件平台上运行 所以如果想要x86跑的系统能够在power cpu也能够运行 需要进行修改 **参考硬件的功能函数并修改操作系统程序代码** 使得改版后能够在另外一个硬件平台运行 称之为**软件移植**）

Windows操作系统本就是针对于个人计算机x86架构的硬件设计的 所以只能在x86计算机运行

不过linux是开源操作系统 所以程序码可以被修改成适合在各种机器上运行的（良好的**硬件控制 资源管理**）

## Unix操作系统

### 兼容分时操作系统

1969年 一个伟大的梦想 bell MIT GE的Multics系统

早期计算机并不是一般人碰到起的 一般都是高科技使用

并且早期的计算机都特别难用 除了指令周期并不快 操作结构也很是困扰 因为那个时候的输入设备之后卡片 阅读机 输出设备只有打印机 用户无法与操作系统互动

后来MIT在1960年发展了所谓的（兼容分时操作系统）可以通过大型主机 通过提供多个终端机以联机进入主机 来利用主机的资源

为了强化Multics的计划



共同发起了Multics计划 想要让大型主机可以提供300个以上的终端机联机的使用目标

不过1969年前后 计划进度落后 资金短缺 贝尔实验室退出了该计划

### Unix出世

贝尔实验室退出Multics计划之后 Ken Thompson已经从中得到一些点子

Thompson因为自己的需要 洗完开发一个小小的操作系统 经过四个月的奋斗 他终于使用汇编语言写出了一组核心程序 同时包括一些核心工具程序 以及一个简单的文件系统

这个文件系统有两个重要的概念

* 所有的程序或者系统装置都是文件
* 不管构建编辑器还是附属文件,所写的程序只有一个目的，且要有效的完成目标

### Unix的正式诞生

1973年，unix正式诞生 Ritchie等人以c语言编写第一个正式的Unix核心

由于贝尔实验室隶属于美国电信大厂AT&T公司 只是AT&T当时忙于其他商业活动 所以对于Unix并不支持也不排斥 此外unix在此期间发展着都是贝尔实验室的工程师 他们对于程序有相当研究 所以Unix在此不容易被一般人所接受

由于unix是以较高阶的c语言写成的 所以相对于汇编语言需要与硬件有密切的配合，高阶的c语言与硬件的相关性就没有那么大了

**1977年 重要的Unix分支 BSD诞生**

虽然贝尔属于AT&T 但是此时对于Unix是采取比较开放的态度，此外Unix是以高阶的C语言写成的，理论上是具有可移植性的

所以 只要取得源代码 并且针对于大型主机的特性加以修订原始码（Source Code）就能够移植到不同的主机上头 所以在1973年以后 Unix便得以与学术界合作开发 最重要的就要属于与加州伯克来大学的合作

柏克莱大学的Bill Joy 取得了Unix的核心原始码后 着手修改成适合自己机器的版本，并且同时增加了很多的工具软件 与 编译软件 并且最终将他命名为 Berkeley Software Distribution(BSD)是Unix很重要的一个分支（后来可以安装在x86架构的FreeBSD就是BSD改版而来的）

### 1979年 重要的System V架构与版权宣告

由于Unix高度移植性和强大的效能 加上当时并没有版权的纠纷 所以很多商业公司开始Unix操作系统的发展例如AT&T 自家的System V IBM的AIX 以及HP 。。。都推出自己的主机搭载自己的Unix系统

但是 我们前面提到的  **操作系统的核心（Kernel）必须要跟硬件配合 以提供及控制硬件的资源进行良好的工作** 而早期每一家生产的计算机硬件的公司还没有所谓**协议**的概念

所以每一家公司生产的硬件自然不会相同 所以他们必须为自己的计算机硬件开发适合的unix系统 所以他们开发的unix操作系统以及内含相关软件并没有办法在其他的硬件架构下工作 另外 并没有厂商针对个人计算机设计Unix 所以**早期并没有支持个人计算机**Unix操作系统出现

如果兼容分时操作系统一般 Unix强调的是多人多任务 但早期286个人计算机架构下的CPU并没有能力达到多任务的 所以没有人打算移植unix到x86上

每一家公司自己推出的unix虽然在架构上大同小异，但是却真的仅仅只能支持自身的硬件 所以 早先的unix只能与服务器（server）或者大型工作站（Workstation）划伤等号

但到了AT&T推出的System V第七版unix之后 开始就可以在x86架构的个人计算机系统安装和运作 不过AT&T由于商业的考虑 于是想将Unix的版权收回去（于是提到了不可以对于学生提供原始码） 同时出现了很多商业纠纷

### 1984年之一 x86架构的Minix操作系统开始撰写并且与两年后诞生

关于1979年的版权声明 Andrew Tanenbaum(谭宁邦)教授很伤脑筋

**既然第七版Unix可以在x86架构上面进行移植 那么是否可以将unix改写移植到x86上 谭宁邦**教授 于是自己动手写了Minix这个Unix like的核心程序 在撰写中 完全没有看Unix核心源码 并且Minux必须能够与Unix兼容

**谭宁邦与1984年开始撰写核心程序到了1986年终于完成 于次年出版相关书籍**

## 1984年 GNU计划与FSF基金会的成立

Richard Mathew Stallman(史托曼)在1984年发起的GNU计划，对于现今的自由软件风潮，有不可磨灭的地位，目前我们的自由或者开源软件，都直接间接受益与GNU计划

#### 一个分享的环境

Richard Mathew Stallman 当时的黑客圈对于软件的着眼点都是 分享，黑客都在相互学习对方的程序代码这样才是产生更优秀代码的程序代码的最佳方式

后来 管理阶层以及黑客群雄们自己的生涯规划等问题 导致实验室人员都离开 ，但史托曼仍然持续在原来的实验室开发新的程序与软件 但他发现一个人并无法完成 于是想要成立一个开放的团体

#### 使用unix开发阶段

1983年 实验室硬件更换 使得史托曼无法继续使用原有的硬件与操作系统继续自由程序的撰写 过去他所使用的Lisp操作系统 是麻省理工的专利软件无法共享，这对于想要成立一个开放的团体的史托曼是一个阻碍

后来 他接触Unix 并发现unix在理论上与实际上 都可以在不同机器间移植。虽然unix依旧是专利软件 但unix架构至少还是开放的 于是他开始转而使用unix系统

于是他开始撰写 unix软件 而且会使可移植形态

#### GNU计划的推展

1984年 史托曼开始GNU计划 **建立一个自由 开放的unix操作系统（Free Unix）**

因为操作系统过于复杂 **于是史托曼就先写可以在unix上面运行的小程序，**而且史托曼绝不看其他软件的原始码，后来一堆人知道免费的GNU软件，于是GNU计划逐渐打开知名度

不论什么软件 都需要编译软件 都需要编译为二进制 如果能够编写一个不错的编译软件（binary program）那不就是大家都需要的软件吗 于是他开始编写c语言编译程序

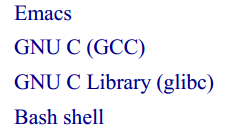
就是相当有名的GNU C Compiler(gcc)

开始撰写gcc并不顺利 他先编写之前的Emacs编辑器写成可以在unix上跑的软件 并公布源代码 当时互联网并未流行 史托曼便借着Emacs以磁带出售 赚了一点钱进而全力撰写其他软件 成立了**自由软件基金会** 请更多的工程师撰写软件 终于完成了GCC

此外还撰写了很多可以被呼叫的(GNU C library) 以及 被使用来操作系统的基本结构BASH shell （都在1990年左右完成）

#### GNU通用公共许可证

1985年 为了避免自由软件被其他人利用称为专利软件 于是他与律师草拟了有名的通用公共许可证（General Public License, GPL）并称呼为copyleft



造成后来很多软件开发者藉由此基础的工具进行程序开发 进一步壮大了自由软件团体

不过 对于GNU（建立一个自由的Unix操作系统）来说 有这些优秀的程序仍无法满足

并没有自由的unix核心

## 1988年 图形接口XFree86计划

鉴于图形用户接口（Graphical User Interface）需求日加严重 在1984年由MIT与其他第三方发表的X window System

并在1988年 成立了非盈利性组织 XFree86这个组织，就是X Window System + Free + x86

而这个XFree86的GUI界面更在Linux的核心1.0版释放出 整合与Linux中

## 1991年 芬兰大学Linux Torvalds一条简讯

1991年 芬兰大学Linux一条消息 以bash gcc编写了一个核心程序

# 关于GNU计划、自由软件与开放原始码

1984年创立GNU计划于FSF基金会的Stallman认为 写程序最大的快乐在于自己发展的良好的软件让大家使用 如果使用方比自己撰写程序的能力强 那么对方修改完程序后并回传修改的程序给自己 那么就可以互相学习

既然程序是想要分享 给大家使用的 那么该程序的原始码就应该要同时释出 这样才能方便大家修改 这个将原始码连通软件程序释放出的举动 在GNU计划范围内就称为自由软件（Free software）运动

## 自由软件的版权

为了避免自己开发出的open source软件被当做专利软件 于是stallman同时将GNU与FSF发展出来的软件 都挂上GPL的版权宣告

FSF核心观念是 版权是促进社会进步的手段 版权本身不是自然权利.

## 自由软件（Free）的真谛

GPL （GNUGeneral public license） 是什么？

自由是一种自由的权利 不是价格

自由软件的重点 并不是免费 而是指自由度的软件 自由度指使用者可以自由的执行 赋值 在发行 学习 修改与强化的自由软件

这样一来 原本你拿到的软件只能在UNIx 上跑 但是经过了原始码的修改 他就可以在linux或者windeow上面来跑 一个软件挂上了GPL版权宣告 他就成了自由软件

注意 ： 所修改的自由软件都不应该 也不能

修改授权 不能将一个GPL授权的自由软件 在修改之后取消GPL授权

单纯贩卖 不能单纯贩卖自由软件（不能单独贩卖自由软件 必须搭配服务）

自由软件与商业行为：目前有很多Linux开发商 为什么他们可以贩卖linux这个gpl授权的软件？ 因为他们大多贩卖售后服务

他们使用的自由软件 都可以在他们网站写在 但是你可以购买光盘 （如果购买了光盘 就会提供数年不等的咨询 售后服务 软件升级 与其他协力工作等附加价值）

## GPL对于自由软件的作者有何优点

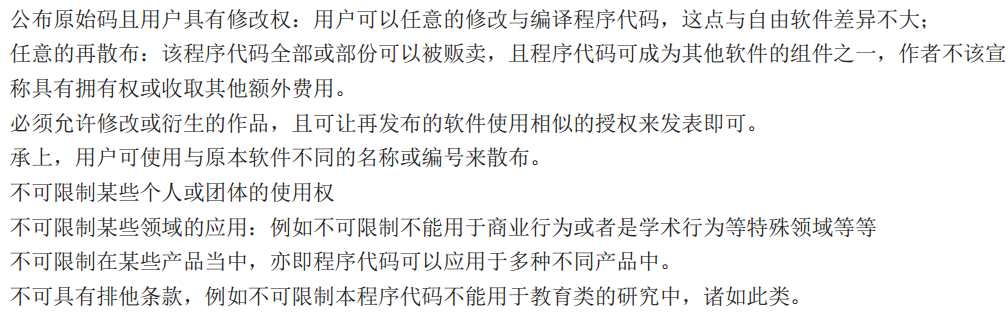
* 软件安全性能较佳
* 软件执行效能较佳
* 软件除错时间较短
* 贡献的原始码永远都在。

注意 授权也会进步 因应原始码分区与重组的问题 与其他开源软件额兼容包容性，以及最重要的数字版权管理（Digital Rights Management,）GPL已经出到第三版 GPLv3

## 开源软件

由于自由软件使用的英文为 free sofrware这个free有免费的意思 所以很多商业公司对于投入自由软件方面确实有些顾虑

为了解决这个问题 1998年成立了【开放原始码促进会 Open Source Initiative】（open source 也可以成为开源软件）



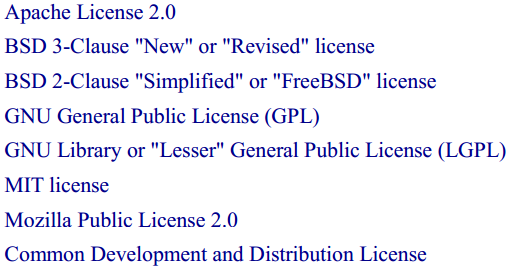
GPL其实也算是开源软件的一个 只是对于商业应用的限制稍微多一点点

与GPL相比 其他开源软件的授权可能比较轻松

* 在发布的授权可以与原本的软件不同
* 开源软件的全部或者部分可以作为其他软件的一部分 无须使用之前相同的授权

其实open source这个名词只是一个指引 在此之前就有好多相关的授权

常见的开放原始码授权有哪些？



GPL 也是合乎open source所定义的授权之一 他更着重保护自由软件本身的学习与发展方向

如果你的软件允许用于商业活动 那么可以考虑 BSD之类的授权

如果你希望商业色彩少一点 那么采用GPLv2授权大概是不二之选

如果允许分支开发 甚至可以采用两种授权

## 专属软件/专利软件（close source）

相对于open source的软件会释放出原始码 close source的程序仅仅退出可执行的二进制程序（binary program） 优点是 有专人维护 不需要更改 但是灵活性降低

一般下还是有免费的专利软件可以使用的 免费的专利软件授权一般为

## Freeware 免费软件

（而非自由软件 free software）

## Shareware

共享软件 一般有试用期限制

# Torvalds的linux发展

Linuxs Torvalds(托瓦兹 1969年出生) 从小就管理一些微计算机 于是他从小就接触到了汇编语言

由于自己的兴趣接触到了unix系统 而早期的计算机只有主机具有运算功能 terminal仅仅负责输入 输出 所以他想到【为什么我不自己搞一部unix来玩？】

不久之后 他就得知 有一个类似unix的系统 并且与unix完全兼容 还可以在intel386上面来跑 就是谭宁邦教授为了教育需要而编写的那个系统 Minix

他在购买了最新的intel386个人计算机知乎 就立刻安装了Minix这个操作系统

## 对于386硬件多任务测试

工作站以上的计算机 特色就是 可以进行多任务的能力

因为理论上一个cpu在一个时间内仅仅能进行一个程序 没关系 如果这个是后具有多任务能力的CPU就会在不同的程序之间切换 假设cpu一面可以工作1Ghz 就是109假设CPU对于每个程序斗志进行100次运作周期 然后就得要切换下一个程序 那么1秒cpu就能够切换106

早期x86芯片对于多任务处理不佳 CPU在不同的工作之间的切换不是很顺畅，但是这个情况在386推出后有很大改善

早期计算机效能没有现在的好 所以压榨计算机效能就成了工程师的一项癖好

一项测试 重要测试是测试386的多功能性

一个程序会一直输出A 一个程序会一直输出B 最后一个程序负责将两个程序进行切换

要达成多任务的环境 除了硬件之外CPU 需要能够具有多任务的特性 操作系统也必须支持这个功能

## 初次释放linux 0.02

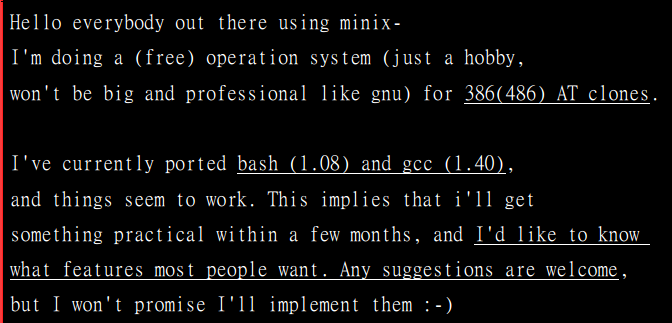
探索完386硬件性能 终于拿到Minix并安装在386上 他发现minix真的很棒

但是谭宁邦教授就是不愿意进行功能的加强 导致一堆工程师 在操作系统功能上面供不应求 【为什么我不自己该写一个我想要的操作系统那？】于是开始核心的撰写

撰写程序需要什么？ 1 能够工作的环境 2 可以将原始码编译为可执行文件的编译程序

好在有GNU 计划提供的bash工作环境 和 gcc编译程序等自由软件

一则简讯



同时为了让自己的linux系统能够兼容unix于是托瓦兹开始将一些能够在unix上面来跑的程序拿来在linux上面跑 不过他发现好多软件无法再Linux核心运作

这个时候他有两种做法 1 修改软件 2 修改Linux 让Linux符合软件能够运作的规范

由于linux希望能够兼容unix所以托瓦兹选择修改linux

于是托瓦兹开始参考POSIX（pou six） 可携式操作系统接口（Portable Operating System Interface）规范 重点规范核心和应用程序之间的接口（主要针对unix与一些软件运行时候的规范）

只要依据这些规范来设计的核心与软件 理论上就可以搭配在一起执行

## Linux的发展 虚拟团队的产生

### 单人维护阶段

首先他将Linux核心放在ftp上面 并请求大家新的版本信息 等用户下载安装之后 如果有问题 或者特殊需求的某些硬件驱动 那么使用者就会汇报给托瓦兹 在处理范围之内 他都能很快处理

### 广大黑客志客加入

总有些硬件无法取得 那么久无法帮助进行驱动程序的编写

有些志工跳出来说 这个硬件我又 我来帮忙撰写驱动程序

托瓦兹来说 他将该驱动程序带入核心 加以测试 只要测试可以运行 并没什么大问题 那么他就能够很乐意的将志工们的程序加入核心

先求有且能跑 再求进一步改进

因为随时都有程序代码加入的情况 于是Linux便逐渐发展成具有模块的功能 将某些功能独立出核心 需要时才加载进入

## 核心功能细部分工发展阶段

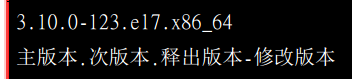
[www.kernel.org](http://www.kernel.org) 网站

这群素未谋面的虚拟团队 在1994年终于完成了Linux核心的正式版 并且加入了x window system的支持 之后发展很是迅速

由于linux是托瓦兹针对于386写的 所以跟硬件关联很大 所以早期Linux确实不具有移植性 但是open source 好处就是可以修改程序代码到适合作业环境

目前 x86 IBM HP 等硬件都支持x86 甚至小型计算机 arm 也大多是arm核心

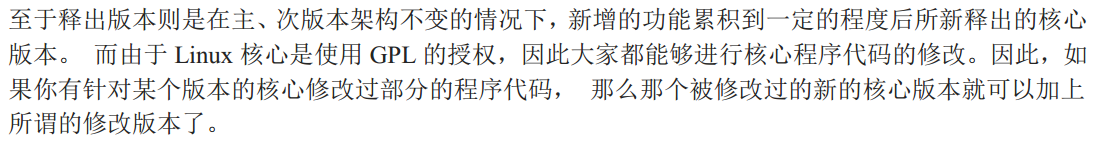
## 核心的版本



在2.6.x 之前

主次版本为奇数 发展中版本

主次版本为偶数 稳定版本



## 主线版本 长期维护版本（longterm version）

不过奇偶编号格式在3.0 后就失效了

3.0 后 核心主要依据主线版本来开发 开发完毕后会往下一个主线版本进行

3.10 就是在3.9 架构下 继续开发出来的版本 新功能的加入 就会有新的主线版本

而旧的版本在新的主线版本出现后 会有两种处理机制

* 结束开发（End of Live eol） 不会继续维护
* 长期维护（longterm） 长期维护版本

## Linux核心版本和linux发布商版本

Linux版本是核心版本 例如 3.10.x-…

### Linux distributions

Linux其实就是一个操作系统最底层的核心以及其提供的核心工具。

他是GUN GPL授权模式的 所以任何人均可以取得原始码与可执行这个核心程序 并且修改 由于Linux参考POSIX设计规范 于是兼容unix操作系统 也称为unix like

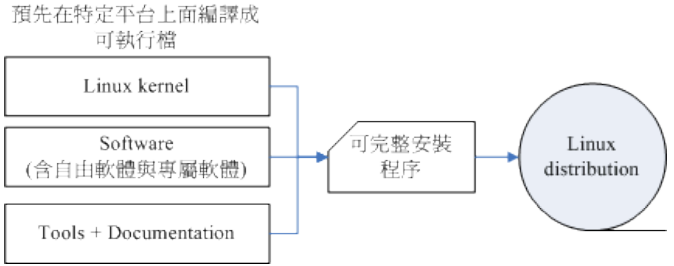
### 可完全安装的Linux发布套件

Linux的出现让GNU计划心里的石头放下了

例如postfix vsftpd apache也都以linux为开发测试平台的计划出现

托瓦兹负责开发的Linux仅仅只有核心和核心提供的工具 不过很多软件可以在Linux上面运作 所以Linux+各种软件就可以完成一个相当完整的操作系统

为了让使用者能够接触Linux 很多商业公司或者非盈利团体 就将Linux核心（含 tools ）与可运行的软件整合起来 加上自己具有创意的工具程序 可以让用户已光盘/DVD/通过网络安装/管理Linux系统 这个kernel + softwares + tools 可完全安装程序 我们称之为linux distribution



Linux 是由黑客工程师写的 要由原始码安装到x86上 这个过程不是谁都会的 一群公司 团体 将Linux配合自由软件 并提供完整的安装程序 值作为光盘 对于一般使用者才具有更大的吸引力

GPL版权宣告 并非不能用来贩卖 而由于Linux的gpl版权宣告 所以商业公司的linux dirtribution通常都可以在internet上面下载 但如果需要服务 可以购买光盘

### 各大Linux distributions的主要异同 支持标准

由于发展Linux的社区和团体很多 有名的 Red Hat SuSe Ubuntu Fedora Debian

那么每一distribution是否都不相同那？

因为每一个都是使用[www.kernel.org](http://www.kernel.org)的核心 而他们选择的软件 都是很有名的

Apache 电子邮件服务器Postfix/sendmail 文件服务器的Samba等等。

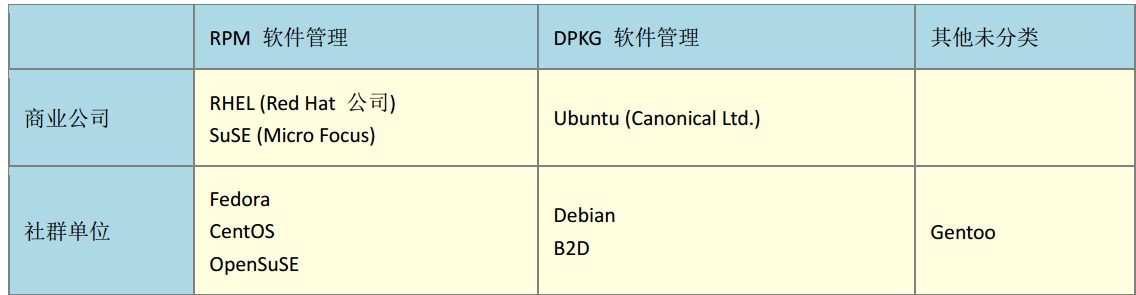
为了让Linux distributions的开发不至太大 且让开发商开发时有所依据

还有Linux Standard Base(LSB) 以及目录架构的File system Hierarchy Standard(FHS)

Distributions主要分为两大系统

一种是使用RPM 安装软件的系统 Red Hat Fedora SuSE

一种是使用Debian 的dpkg方式安装软件的系统 Debian Ubuntu B2D



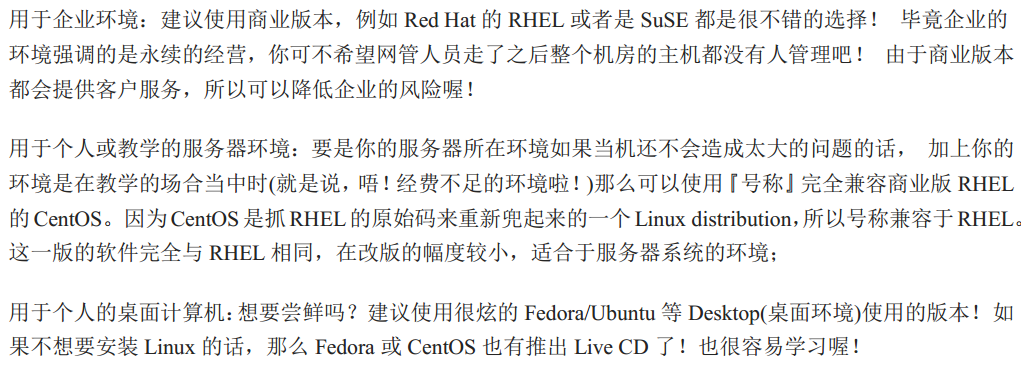
### CLE 计划 是Linux对于中国的延伸

## 选择适合的Linux distribution

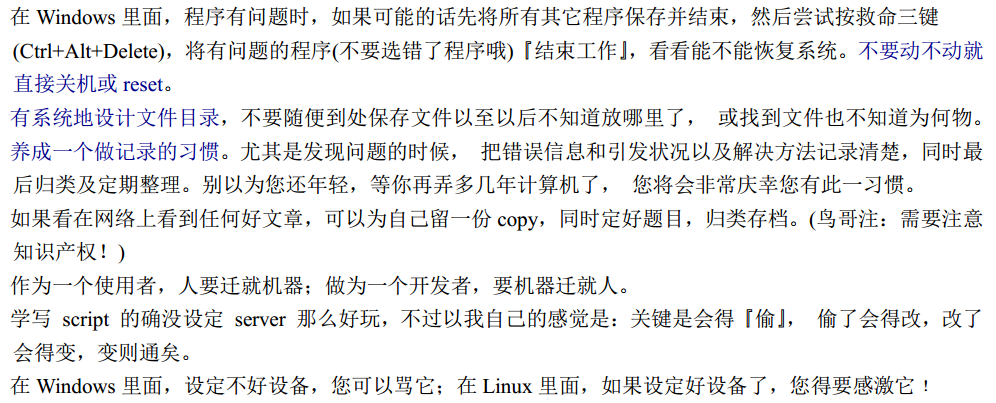
套件管理方式主要分为 Debian 的dpkg和redhat系统的RPM方式

RPM centOS 使用较多

如果想要追求更加严谨的Linux版本 Debian不错 如果效能至上 或者Gentoo捕获







# 重点回顾



# 习题回顾

