



کسب و کار و اقتصاد

لینوکس و متن باز

نویسنده:
مارتین فینک
مدیر بخش متن باز شرکت HP

ترجمه:
دکتر محمد خوانساری
مهندس هادی محمدی
مهندس فرشته پورامینی

ویراسته:
مهندس زهرا احمدی
مهندس آرش معبدی

فهرست برگه

فینک، مارتین

کسب و کار و اقتصاد لینوکس و متن باز /نویسنده مارتین فینک، ترجمه محمد خوانساری.هادی محمدی.فرشته پورامینی. ویراسته زهرا احمدی، آرش معبدی. با همکاری مرکز تحقیقاتی فناوری اطلاعات و ارتباطات پیشرفته دانشگاه صنعتی شریف--تهران:شورای عالی انفورماتیک کشور، دبیرخانه، ۱۳۸۴.

۱۹۰ ص. مصور، جدول

ISBN:964-96535-5-4

فهرست نویسی براساس اطلاعات فیبا

عنوان اصلی: The business and economics of Linux and open source. c2003

۱. بازارگانی— برنامه‌های کامپیوتری. ۲. نرمافزار متن‌باز. ۳. سیستمهای عامل (کامپیوتر). ۴. منابع شبکه کامپیوتری. ۵. سیستم عامل لینوکس. الف. خوانساری، محمد، مترجم. ب. محمدی، هادی، مترجم. ج. پورامینی، فرشته مترجم. د. دانشگاه صنعتی شریف، مرکز تحقیقاتی فناوری اطلاعات و ارتباطات پیشرفته. ه. شورای عالی انفورماتیک کشور، دبیرخانه. و. عنوان

۰۰/۵

HF ۵۵۴۸/۴/۹

۱۳۸۴

۱۷-۴۱۰۱۷

کتابخانه ملی ایران

کسب و کار و اقتصاد لینوکس و متن باز

ناشر: دبیرخانه شورای عالی انفورماتیک

ترجمه: دکتر محمد خوانساری- مهندس هادی محمدی- مهندس فرشته پورامینی

ویراستار فنی: مهندس زهرا احمدی

ویراستار ادبی: مهندس آرش معبدی

حروفچینی: پریسا سمالی- لیدا زلکی

صفحه آرایی: محمدمهدی بزرگار

ناظر چاپ: مریم مسعودی

طرح روی جلد: مهندس بهروز عبادی

چاپ و صحافی:

تیراژ: ۱۵۰۰ نسخه

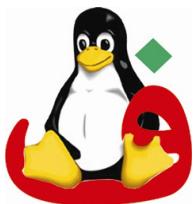
قیمت: ۲۰۰۰۰ ریال

کلیه حقوق، متعلق به شورای عالی انفورماتیک می‌باشد. اجازه تکثیر، توزیع و یا تغییر این اثر تحت شرایط اجازه نامه مستندات آزاد گنو (که توسط بنیاد نرمافزارهای آزاد تهیه گردیده) داده می‌شود.

شابک: ۹۶۴-۹۶۵۳۵-۵-۴

(P)

سازمان مدیریت و بنامه ریزی کشور
شورای عالی انفورماتیک کشور



طرح ملی نرم افزارهای آزاد/متن باز

(گنو/ لینوکس فارسی)

با همکاری مرکز تحقیقاتی فناوری اطلاعات و ارتباطات پیشرفته
دانشگاه صنعتی شریف
و حمایت دبیرخانه شورای عالی اطلاع رسانی

کلیه حقوق، متعلق به شورای عالی انفورماتیک می باشد.
اجازه تکثیر، توزیع و یا تغییر این اثر تحت شرایط اجازه نامه
مستندات آزاد گنو (که توسط بنیاد نرم افزارهای آزاد تهیه
گردیده) داده می شود.

امروزه فناوری اطلاعات به عنوان یکی از مهمترین زیرساختهای توسعه در کشورهای دنیا شناخته شده است. رشد روزافزون این فناوری در کشورهای توسعه یافته، شکاف دیجیتال بین این کشورها و کشورهای در حال توسعه را افزایش می‌دهد.

یکی از حوزه‌هایی که در رشد فناوری اطلاعات در دنیا تاثیر بسزایی داشته، مقوله نرمافزارهای آزاد/متن باز است. جنبش نرمافزارهای آزاد/متن باز پس از ۲۰ سال تلاش برای آزادی نرمافزار در سراسر دنیا امروزه به رشد و بالندگی رسیده است و باعث پیشرفت و تحولی عمیق در حوزه فناوری اطلاعات شده است.

کشورهای اتحادیه اروپایی، چین، راپن، بربل، آرژانتین، پرو، آفریقای جنوبی و حتی افغانستان برنامه‌های مدونی برای بکارگیری و توسعه این نرمافزارها برای نیل به اهداف خود اعلام کردند، کسانی که به این نرمافزارها به دیده تردید می‌نگریستند، پی به اهمیت آن در سیاست‌گذاری توسعه فناوری اطلاعات در کشورها بردند. این‌گونه سیاست‌گذاری نیازمند همکاری و هماهنگی ارکان مختلف دولت در راستای تحقق آنهاست.

در ایران نیز این حرکت جهانی در قالب طرح ملی نرمافزارهای آزاد/متن باز (گنو/لینوکس فارسی) از حدود سه سال قبل با کارفرمایی دیپرخانه شورای عالی انفورماتیک و مدیریت مرکز تحقیقاتی فناوری اطلاعات و ارتباطات پیشرفتنه دانشگاه صنعتی شریف و حمایت دیپرخانه شورای عالی اطلاع رسانی برای تولید جایگزین نرمافزارهای مهم و کاربردی داخل کشور بر مبنای بومی‌سازی نرمافزار آزاد و ایجاد تنوع نرمافزاری شروع شده است. این طرح بستر ساز تولید سامانه عامل ملی کشور است که می‌تواند به خوبی به عنوان جایگزین سامانه‌عامل ویندوز استفاده شود. در حال حاضر با توجه به فعالیت انجام گرفته نسخه‌های اولیه جایگزین برای سامانه‌عامل، برنامه‌های دفتری و بانکهای اطلاعاتی تا حد خوبی انجام شده است، که این تلاش در جهت استقلال و خودکفایی کشور در صنعت نرمافزار قابل تقدیر است.

با توجه به جوان بودن این حرکت در کشور لزوم فرهنگ‌سازی و تولید محتویات آموزشی ضروری به نظر می‌رسد. دیپرخانه شورای عالی انفورماتیک بر خود واجب می‌داند که تا حد امکان بستر لازم برای گسترش این فعالیت را آماده نماید. در همین راستا این شورا اقدام به تهیه مجموعه کتابهایی با عنوان «مجموعه نرمافزارهای آزاد/متن باز» – با پوشش دادن طیف کلی از مخاطبین این حوزه مانند مدیران، کارشناسان رسته فرابری داده‌ها، کاربران نهایی، دانشجویان، توسعه‌دهندگان و برنامه‌نویسان – نموده است که کتاب حاضر نیز از همین مجموعه است. امید است این مجموعه کتابها بتواند کمکی در جهت بالابردن آگاهی عمومی جامعه در حوزه نرمافزارهای آزاد/متن باز شود.

دیپر شورای عالی انفورماتیک کشور بهمن ۸۴

فهرست

۱	یادداشت نویسنده
۴	بخش اول اصول پایه
۵	فصل اول کسب و کار لینوکس و متن باز
۵	۱-پذیرش لینوکس
۶	۱-۲-نگاهی گذرا به لینوکس و واژگان متن باز
۸	۱-۳-حجم کاری لینوکس
۹	۱-۴-مزایای تجاری
۹	۱-۴-۱-هزینه
۱۰	۱-۴-۲-در دسترس بودن مطالب آموزشی
۱۱	۱-۴-۳-پشتیبانی
۱۱	۱-۴-۴-توسعه نرم افزار
۱۲	۱-۴-۵-ارتقاء
۱۲	۱-۵-موانع رشد لینوکس
۱۲	۱-۵-۱-در دسترس بودن برنامه های کاربردی
۱۳	۱-۵-۲-خطر تجاری
۱۳	۱-۶-نقش آفرینان متن باز
۱۷	۱-۷-جمع بندی
۱۸	فصل دوم لینوکس - قلب سامانه عامل
۱۹	۱-۲-سامانه عامل
۲۰	۱-۲-۲-هسته لینوکس
۲۱	۱-۲-۲-۱-طراحی هسته
۲۲	۱-۲-۲-۲-پیمانه های هسته لینوکس
۲۳	۱-۲-۲-۳-وصله های هسته لینوکس
۲۴	۱-۲-۲-۴-تکثر هسته
۲۵	۱-۳-۲-طراحی هسته لینوکس و کنترل نسخه ها
۲۸	۱-۳-۱-پشتیبانی چند بستر
۲۹	۱-۳-۲-بازار بزرگ در مقابل پردازنده های نامشهور

۳۰	۴-۲-لینوکس روی میز کار.....
۳۱	۴-۵-مقیاس پذیری عمودی و افقی.....
۳۲	۶-۲-لینوکس توکار.....
۳۳	۷-۲-جمع بندی.....
۳۴	فصل سوم متن باز - هدایت مسیر قانونی به آزاد بودن.....
۳۴	۱-آزادی متن باز بودن.....
۳۵	۲-تعريف متن باز.....
۳۷	۳-۳-مالکیت معنوی و دوطرفه بودن.....
۳۹	۴-۳-مالکیت حق نشر و مجوز دوگانه.....
۴۰	۵-۳-مجوزها - متن باز و غیر متن باز.....
۴۰	۵-۱-مجوزهای متن باز.....
۴۳	۵-۲-LGPL و GPL.....
۴۶	۵-۳-GPL و پیمانههای هسته لینوکس.....
۴۷	۵-۴-LGPL و وراثت کلاس.....
۴۸	۵-۵-مجوزهای غیر متن باز.....
۴۹	۶-۳-صادر کردن و رمزگاری.....
۵۰	۷-۳-متداولزی توسعه متن باز.....
۵۰	۷-۱-مجوز.....
۵۱	۷-۲-روشها.....
۵۲	۷-۳-صفات.....
۵۵	۸-۳-جمع بندی.....
۵۶	فصل چهارم انجمنها و شرکتها.....
۵۶	۱-۴-لینوکس.....
۵۶	۱-۱-هسته.....
۵۷	۱-۲-پردازندها.....
۵۸	۱-۳-سامانههای پرونده.....
۵۹	۱-۴-پشتیبانی سخت افزار.....
۵۹	۲-۴-خدمات وب و کارسازهای برنامههای کاربردی.....
۶۲	۳-۴-زبانهای برنامهنویسی.....
۶۵	۴-۴-محیطهای رومیزی و برنامههای اداری.....

۶۹۴-۵-پایگاه داده‌ها
۷۰۴-۶-PDA
۷۰۴-۷-خوشها
۷۱۴-۸-سازمانها
۷۳۴-۹-جمع بندی
۷۴۶-۷-بخش دوم لینوکس در عمل
۷۵۶-۸-فصل پنجم توزیعها-تکمیل لینوکس
۷۵۶-۹-۱-توزیع لینوکس
۷۷۶-۱۰-۲-بسته‌های نرم‌افزاری
۸۰۶-۱۱-۱-قالب بسته
۸۰۶-۱۲-۳-عرضه کنندگان توزیعها
۸۱۶-۱۳-۱-توزیعهای فرآگیر
۸۴۶-۱۴-۲-۳-توزیعهای محلی
۸۴۶-۱۵-۳-۳-توزیعهای اختصاصی
۸۵۶-۱۶-۴-توزیعهای سامانه‌های عامل غیرلینوکس
۸۵۶-۱۷-۵-طراحی توزیع برای خودتان
۸۷۶-۱۸-۶-پشتیبانی از چندین توزیع
۸۷۶-۱۹-۷-استانداردها
۸۸۶-۲۰-۸-۱-جمع بندی
۸۹۶-۲۱-۹-۱-هزینه‌ها
۸۹۶-۲۲-۹-۲-پذیرش یک رویکرد ناقص
۹۶۶-۲۳-۹-۳-فراهم کردن لینوکس و نرم‌افزارهای متن‌باز
۹۸۶-۲۴-۹-۴-تغییردادن نرم‌افزار متن‌باز
۹۹۶-۲۵-۹-۵-جمع بندی
۱۰۰۶-۲۶-۱۰-۱-فصل هفتم استانداردها - یک لینوکس
۱۰۰۶-۲۷-۱-چرا استاندارد؟
۱۰۱۶-۲۸-۲-۲-گروه استانداردهای آزاد
۱۰۲۶-۲۹-۳-۳-پایه استانداردهای لینوکس (LSB)
۱۰۳۶-۳۰-۴-۴-۱-اجزای توزیعهای LSB

۱۰۷	۵-۷- برنامه‌های منطبق بر LSB
۱۰۷	۶-۷- آینده LSB
۱۰۷	۷-۷- بین‌المللیسازی لینوکس
۱۰۸	۸-۷- آزمایش و تطابق
۱۰۹	۹-۷- توزیعهای تخصصی لینوکس
۱۰۹	۱۰-۷- جمع‌بندی
۱۱۰	فصل هشتم عملیات: بکارگیری لینوکس و متن باز
۱۱۰	۱-۸- پیاده سازی
۱۱۱	۲-۸- پیاده سازی جغرافیایی
۱۱۱	۳-۸- مهاجرت و هم‌زیستی
۱۱۲	۴-۸- ساخت افزار
۱۱۳	۵-۸- اطلاعات
۱۱۳	۱-۵-۸- Endian
۱۱۴	۲-۵-۸- سامانه‌های پرونده
۱۱۵	۳-۵-۸- به اشتراک گذاری
۱۱۶	۶-۸- مدل‌های برنامه نویسی
۱۱۷	۷-۸- برنامه‌های کاربردی
۱۱۷	۸-۸- محیط‌های رومیزی
۱۱۸	۹-۸- تهیه مجوز و خرید برنامه‌ها
۱۱۸	۱۰-۸- پشتیبانی
۱۱۹	۱۱-۸- هر نرم‌افزار یک شرکت
۱۲۰	۱۲-۸- تمامی نرم‌افزارها، یک شرکت
۱۲۰	۱۳-۸- استفاده از پشتیبانی جامعه لینوکس و متن باز
۱۲۱	۱۴-۸- تأثیرگذاری و ارتباطات
۱۲۱	۱۱-۸- آموزش
۱۲۳	۱۲-۸- جمع‌بندی
۱۲۴	بخش سوم نقش متن باز در کسب و کار
۱۲۵	فصل نهم بازار شرکتی
۱۲۵	۱-۹- کانترال و بازار
۱۲۶	۲-۹- ساختار، سیاستها را دنبال می‌کند

۱۲۸	۳-۳-بازار ساخت یافته.....	۹
۱۳۰	۴-۴-معاونت مهندسی نرم افزار	۹
۱۳۱	۵-۵-تیم فناوری.....	۹
۱۳۱	۱-۵-۱-معمار اولیه.....	۹
۱۳۳	۲-۵-۲-معمار عرضه.....	۹
۱۳۴	۳-۵-۳-معماران زیرسامانهها.....	۹
۱۳۵	۴-۵-۴-مهندسان.....	۹
۱۳۶	۶-۶-۶-منابع انسانی.....	۹
۱۳۶	۷-۶-۷-۱-رشد استعدادها.....	۹
۱۳۶	۸-۶-۸-۲-مدیریت افراد (مهارت‌ها).....	۹
۱۳۷	۹-۶-۹-آموزش	۹
۱۳۷	۱۰-۷-۱۰-آزمایش و یکپارچگی	۹
۱۳۸	۱۱-۷-۱۱-۱-گروه مالی	۹
۱۳۸	۱۲-۸-۱۲-بازاریابی	۹
۱۳۹	۱۳-۹-۹-بیر به بازار	۹
۱۴۰	۱۰-۹-۱۰-سایر اجزای ساختاری	۹
۱۴۰	۱۱-۹-۱۱-جوامع دروازه‌ای	۹
۱۴۱	۱۲-۹-۱۲-خطرات و مشکلات	۹
۱۴۲	۱۳-۹-۱۳-جمع‌بندی	۹
۱۴۳	فصل دهم ارزش محصول به عنوان تابعی از زمان.....	
۱۴۳	۱-۱۰-۱۰-صنعت داروسازی.....	
۱۴۴	۱-۱۰-۱۰-۱-هزینه، ارزش، بازگشت سرمایه و زمان.....	
۱۴۶	۱-۱۰-۱۰-۱-۲-بازیابی از شرایط عمومی شدن محصول	
۱۴۷	۱-۱۰-۱۰-۲-تأثیر متن باز بر نرم افزار.....	
۱۴۹	۱-۱۰-۳-کاهش ارزش به عنوان یک مزیت رقابتی.....	
۱۵۱	۱-۱۰-۴-۴-ارزش در حصار زمان.....	
۱۵۲	۱-۱۰-۵-۵-کلاسهای بنیادی Pilot	
۱۵۲	۱-۱۰-۶-Jump-۶-Jump	
۱۵۵	۱-۱۰-۷-۷-جمع‌بندی	
۱۵۶	فصل یازدهم مدل‌های تجاری درآمدزا.....	

۱۱-۱-ارزش محصول خود را بدانید.....	۱۵۶
۱۱-۲-نرم افزار تجاری و لینوکس.....	۱۵۷
۱۱-۳-پشتیبانی و خدمات نرم افزارهای متن باز.....	۱۵۸
۱۱-۴-ترکیب و بالا بردن ارزش.....	۱۵۹
۱۱-۵-تجاری سازی با یک مجوز دوگانه.....	۱۶۰
۱۱-۶-سخت افزار.....	۱۶۲
۱۱-۷-تمایز بین دستگاه واسط و کاراییهای آن.....	۱۶۲
۱۱-۸-مستندات باز برای واسط سخت افزاری.....	۱۶۴
۱۱-۹-همراه کردن نرم افزارهای مورد نیاز با سخت افزار.....	۱۶۴
۱۱-۱۰-مدل پایان چرخه زندگی.....	۱۶۵
۱۱-۱۱-ایجاد یک اکوسیستم.....	۱۶۵
۱۱-۱۲-جمع بندی.....	۱۶۸
۱۲-۱-فصل دوازدهم پیاده سازی متن باز در کسب و کار.....	۱۶۹
۱۲-۲-عرضه نرم افزار متن باز.....	۱۷۰
۱۲-۳-مسائل تجاری.....	۱۷۰
۱۲-۴-۱-پیاده سازی.....	۱۷۵
۱۲-۴-۲-بازاریابی.....	۱۷۷
۱۲-۴-۳-نگهداری.....	۱۷۸
۱۲-۴-۴-متن باز داخلی.....	۱۷۹
۱۲-۴-۵-تریبیت مهندسان.....	۱۸۰
۱۲-۵-۱-تصمیم گیریهای مربوط به سازمان.....	۱۸۱
۱۲-۵-۲-تعیین نوع مجوز.....	۱۸۱
۱۲-۵-۳-دیوار آتش.....	۱۸۲
۱۲-۵-۴-توسعه فناوری اطلاعات.....	۱۸۲
۱۲-۵-۵-جبران زیان.....	۱۸۲
۱۲-۶-۱-فصل سیزدهم منابع انسانی: استفاده بهترین استعدادها.....	۱۸۴
۱۲-۶-۲-قراردادهای استخدامی.....	۱۸۴
۱۲-۶-۳-سیاستهای کار در پروژه.....	۱۸۵
۱۲-۶-۴-استخدام شخص مناسب.....	۱۸۵

۱۸۶	۱-۳-۱۳-فناوری
۱۸۶	۲-۳-۱۳-خانه جامعه
۱۸۶	۳-۳-۱۳-نگهدارنده یا برنامه‌نویس
۱۸۷	۴-۳-۱۳-شهرت و احترام در جامعه
۱۸۷	۵-۳-۱۳-تعامل اینترنتی
۱۸۸	۶-۳-۱۳-کارهای عرضه شده
۱۸۸	۷-۳-۱۳-جغرافیا
۱۸۹	۸-۳-۱۳-مراحل را بشمارید
۱۸۹	۹-۳-۱۳-ساختاربندی تیمهای
۱۸۹	۱۰-۳-۱۳-استخدام راهبران بر جسته
۱۹۰	۱۱-۳-۱۳-جمع‌بندی

یادداشت نویسنده

اصطلاح «نیاز مادر اختراع است» بسیار درست می‌باشد؛ زیرا در صورتی که موضوع دیگری در این زمینه موجود بود، هرگز این پروژه را آغاز نمی‌کرد. البته این روزها اطلاعات فراوانی در زمینه‌های متعدد در اینترنت موجود است، اما نسخه‌ای که شامل موضوعات لینوکس و متن‌باز به صورت توانمند باشد، نیافتند. همچنین بهنظر من برخی از این نمونه‌های متن‌باز، تاکنون برای مدیران صنفی به ثبت نرسیده‌اند.

نهضت لینوکس و متن‌باز موجب تحریک «جنگهای مذهبی» میان گروههای مختلف گردید که با موضع‌گیریهای افراطی و اغلب غیرواقع‌گرایانه همراه بود. این کتاب نه در مورد جنگ و نه درباره علل آن است؛ بلکه تنها در زمینه تجارت بوده و به تعامل لینوکس و متن‌باز و همچنین حضور فraigیر آنها در صنعت فناوری برتر صحه می‌گذارد. گرچه، یک اعتراف ساده تنها به‌منظور کنارآمدن با مسائل بنیادی که توسط لینوکس و متن‌باز ایجاد شده‌اند، کافی به‌نظر نمی‌رسد.

چه کسانی می‌توانند مخاطب این کتاب باشند؟

این کتاب مستقیماً مدیران تجاری را مخاطب قرار داده است. تعدادی از شما ممکن است در زمرة مدیران فناوری اطلاعات (IT) باشید که سعی در بررسی ارزشهای لینوکس و متن‌باز در صنعت دارند. به عبارت دیگر، چرا شما قصد ایجاد تغییراتی دارید؟ دیگر مدیران ممکن است نرمافزار را به منظور استفاده داخلی یا فروش تجاری تولید کنند. در چنین مواردی، نهضت متن‌باز به‌عنوان راهی برای اعمال نفوذ در خیل عظیم تولیدکنندگان به کارگرفته می‌شود، اما پیداکردن بهترین راه ممکن برای پذیرش در میان جامعه بسیار دشوار بوده و هنوز هم سودهای کلانی برای سرمایه‌گذاران به ارمغان می‌آورد. این کتاب برای مدیران کاملاً مناسب بوده و مدیریتی سودآور را در دو سطح بررسی می‌کند. چنانچه شما از جمله مدیران اجرایی هستید، این کتاب شما را در هدایت گروه‌تان و یافتن راه حل‌های مناسب یاری نموده و هنگامی که گروه پیشنهادهایی در رابطه با لینوکس و متن‌باز ارائه می‌دهد، کمک می‌کند راه کار مناسب‌تر را اختیار کنید. در صورتی که شما جزء رتبه‌های متوسط مدیریتی می‌باشید، این کتاب به شما کمک می‌کند تا نسبت به درستی برنامه‌ها و طرح‌هایتان در رابطه با مدیریت پیشرفته و همچنین اختیار الگوهایی جدید و تجاری مربوط به لینوکس و متن‌باز اطمینان حاصل نمایید.

هدف این کتاب، تولیدکنندگان نمی‌باشد؛ به دنبال یافتن راهی برای هدایت هرگونه پیمانه‌های کد برای نرم‌افزارهای تحت لینوکس و یا هر پروژه متن‌باز دیگری نیز نمی‌باشد. مواردی نیز وجود دارد که تولیدکنندگان غیرحرفه‌ای به دنبال یافتن نقطه‌نظری در مورد جوامع لینوکس یا متن‌باز هستند که این کتاب تا حدودی می‌تواند مفید واقع شود.

در چندسال گذشته، من همه روزه در این زمینه جدید و جالب تجاری مشغول به کار بوده‌ام. آغاز کاری که حاوی مضامینی متفاوت بوده و برخی از قواعد بنیادی آن در حال تغییر می‌باشند. همواره با دشواریهایی همراه است. برای بسیاری افراد، مردود دانستن این‌گونه تغییرات و همچنین تلاش برای ادامه وضع کنونی بسیار طبیعی می‌نماید؛ و من همه روزه در حال سازش با

این مقاومتها بوده‌ام، درحالی که مضامین تجاری در ارتباط با لینوکس و متن‌باز هنوز بسیار جدی هستند، مدیران صنفی دریافت‌هایند که چشم‌پوشی از این رویدادها به صلاح‌شان نمی‌باشد. جالب اینجاست که اقتصاد بازار به مضامین جدید تجاری از قبیل لینوکس و متن‌باز بسیار نیاز دارد و پس از تثیت این موارد، کسانی که به این مسائل بی‌توجه بوده‌اند، بازنه‌اصلی خواهند بود. رقیانی که این مضامین را پذیرفته و به شدت در حال بهره‌برداری از این فرصت‌های جدید هستند، درواقع در دست گرفتن کنترل تجارت و کنارزدن رقیانی را آغاز نموده‌اند که نسبت به این موارد بی‌تفاوت بوده‌اند. اگرچه کسانی که به سرعت به سمت منابع بی‌صرف و موقت سوق پیدا می‌کنند، سرانجام از دور رقابت خارج شده و یا اجباراً، موضع خود را به سمت یک تجارت پذیرفته شده، تغییر می‌دهند. افرادی هستند که هنوز لینوکس و متن‌باز را در زمرة منابع موقت می‌دانند. هم اینک سامانه‌عامل لینوکس بیش از ۱۰ سال دارد. نهضت متن‌باز که به همراه نهضت نرم‌افزار آزاد آغاز شد، در حال نزدیک شدن به بیستمین سال موجودیت خود می‌باشد. واضح است که تصور ناکامی آن بسیار بعید بوده و اینک کسب و کارها نیاز به کنار آمدن با این واقعیت‌های جدید دارند.

این کتاب چگونه سازمان یافته است؟

بخش یک این کتاب درآمدی بر جهان لینوکس و متن‌باز می‌باشد. فصل اول با بررسی بر روی دلایل بنیادی تجاری آغاز می‌شود که چرا این نهضت جدید برای تجارت مناسب است و چگونه ارزش خود را نشان می‌دهد. همچنین یک دید پایه‌ای از اصطلاحات متداول در این زمینه می‌دهد که می‌توانید با کمک آن بقیه کتاب را دنبال کنید. فصل دوم به صورت کاملاً عمقی به بررسی هسته لینوکس می‌پردازد. این هسته ممکن است یک موضوع کاملاً فنی در تجارت محسوب شود. امروزه، درک چگونگی تناسب مؤلفه‌های^۱ آن بسیار ضروری به نظر می‌رسد. فصل بعدی مختصرًا به شرح تعریف متن‌باز می‌پردازد. توجه به مجوزهای متن‌باز نیز یکی از مؤلفه‌های پایه‌ای آموزشی می‌باشد. بخش یک با نگاهی عمیق به تعدادی از جوامع و سازمانهایی که به منظور پیوستن به این نهضت به آشنایی با آن نیاز دارند، به پایان می‌رسد.

بخش دوم نگاهی به سامانه عامل لینوکس دارد. این بخش با مروری بر توزیعهای لینوکس- به منظور کمک به شناساندن چگونگی ترکیب هسته لینوکس با دیگر قسمت‌هایی که سامانه عامل را دربرمی‌گیرند- آغاز می‌شود. سپس، تحلیلی کامل از سنجش هزینه‌های IT در رویکرد متن‌باز، ارائه می‌دهد؛ همچنین بررسی می‌کند که چگونه استانداردهای کلیدی بر روی این هزینه‌ها تأثیرگذارند و کدام یک برای موفقیت آتی لینوکس اهمیت دارند. در آخر، مروری بر عملکردهایی از طرز استفاده، انتقال و همزیستی به منظور پشتیبانی و آموزش خواهیم داشت.

¹ components

بخش آخر این کتاب به طور کامل به بحث و بررسی در مورد تأثیرات متن باز بر شرکتهای حقوقی و تولیدکنندگان نرم افزارهای تجاری می پردازد. در این بخش نمونه های جدید و بنیادی تجاری مورد بررسی قرار می گیرند؛ نگاهی به چگونگی ساختار جامعه متن باز، بعضی عناصر فرهنگی، و چگونگی قیاس آن با ساختارهای حقوقی خواهیم داشت. همچنین ارزشهاي فراوان تأثیر متن باز در طول زمان ارائه خواهد شد، که شما را برای مبحث بعدی آماده خواهد کرد و بررسی جامعی از نمونه های تجارت متن باز و چگونگی کسب درآمد از این طریق خواهد بود. دو فصل آخر فرآیند ترکیب متن باز با شرکتهای دیگر و اثرات مدیریت افراد بر کار با این جامعه نوپای تولیدکنندگان را مورد بحث و بررسی قرار می دهد.

بسیاری از مضماین ارائه شده، بهویژه در بخش^۳، جدید می باشند. اطلاعات، به طور امیدوارکننده ای، شما را در پیاده سازی پروژه های متن باز در شرکت تان و همچنین در برقراری روابط همکاری و توافقی با جامعه بزرگ تولیدکنندگان یاری خواهند کرد.

همان طور که کار با متن باز را آغاز نموده اید و می خواهید دریابید که متن باز واقعاً چیست، به شما توصیه می کنم که این کتاب را به عنوان یک پروژه متن باز حتماً بخوانید. امیدوارم که هریک از شما در این زمینه به مقاله نویسانی متبحر و صاحب نظر تبدیل شوید. این مقاله ناتمام به شما ارائه می شود و امید است که شما نیز نظرات سازنده خود را- بیش از آنها بیکاری که ما را در فرآیند بازبینی یاری کردند- با ما در میان بگذارید تا به یک پروژه عالی تبدیل شود.

مارتین فینک

بخش اول اصول پایه

به دنیای لینوکس و متن‌باز خوش آمدید. برای خواندن این کتاب داشتن یک پیش زمینه فنی لازم است. اما نیازی به داشتن دانش قبلی در مورد لینوکس و متن‌باز نیست. در قسمت اول مفاهیم پایه‌ای در مورد سامانه‌عامل متن‌باز، مدل توسعه و مراحل مجوزگیری آمده است. فصل اول دلایل نیاز به لینوکس و متن‌باز را از لحاظ تجاری بررسی می‌کند. فصل دوم هسته لینوکس را به طور مفصل شرح می‌دهد. فصل سوم به تعریف متن‌باز و مجوزهای مربوط می‌پردازد و در نهایت فصل چهارم در مورد سرعت رشد و پیشرفت متن‌باز دیدی کلی ارائه می‌دهد.

فصل اول

کسب و کار لینوکس و متن باز

لینوکس و متن باز فناوریها، ایده‌ها، مفاهیم و نمونه‌های بسیاری را شامل می‌شوند. لینوکس، سامانه‌عامل جدید و عظیمی است که با استفاده از متداول‌تری متن باز توسعه یافته است. برای درک لینوکس و فهم تأثیر توسعه و متن باز بر روی کسب و کار و سازمان، نگاهی گذرا به برخی مفاهیم امری ضروری است. این فصل قصد دارد شما را با مفاهیم زیر آشنا کند:

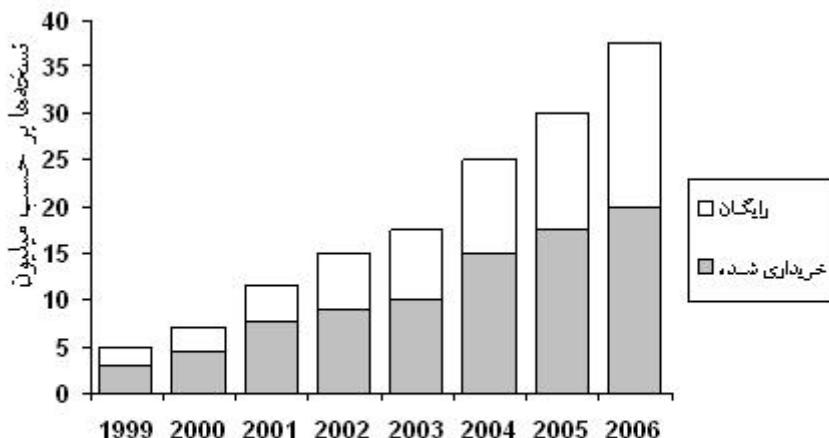
- لینوکس کجا استفاده می‌شود و نحوه رشد آن چگونه است.
- واژگان فنی که در فهم این کتاب مفید خواهد بود.
- مزایای اصلی تجارت لینوکس و متن باز.
- موانع اصلی محدود کننده رشد لینوکس.
- نقش آفرینان اصلی در جامعه متن باز.

این فصل اهمیت لینوکس و متن باز را در تجارت بررسی می‌کند و مفاهیم پایه‌ای - که آشنایی با آنها برای خواندن این کتاب لازم است - را شرح می‌دهد.

۱-۱- پذیرش لینوکس

به طور معمول در بررسیهای مختلف، از نمودارهایی که نمایانگر بازده مورد نظر یک محصول یا خدمات در بازار مورد هدف هستند، استفاده می‌شود. این کار در مورد لینوکس نمی‌تواند خیلی دقیق باشد، چرا که محاسبه دقیق تعداد نسخه‌هایی از لینوکس که در حال استفاده هستند، غیر ممکن است. در واقع این محاسبه برای لینوکس به واسطه متن باز بودن آن بسیار دشوار است. هر چیز که امکان خرید توزیعهای لینوکس وجود دارد، ولی شرکتها اغلب یک نسخه از لینوکس می‌خرند، آن را مطابق با نیازهای خود تغییر می‌دهند و سپس آن را به کار می‌گیرند. از طرف دیگر لینوکس را می‌توان از منابع مختلف به طور رایگان دریافت کرد. لینوکس در لوح فشرده همراه با برخی کتابهای محصولات دیگر نیز عرضه می‌شود. همچنین برخی از کاربران برای به کارگیری چندین سامانه‌عامل مختلف روی یک رایانه، سامانه‌های خود را به صورت بوت دوگانه پیکربندی می‌کنند.

این موارد شمارش دقیق تعداد لینوکسهای مورد استفاده را دشوار می‌کند. در عین حال تحلیل‌گران زیادی مثل IDC نمودارهایی مانند نمودار شکل ۱-۱ را برای داشتن یک تقریب منتشر کرده‌اند. شکل ۱-۱ تعداد نسخه‌های لینوکس - هم رایگان و هم خریداری شده - را نشان می‌دهد.



شکل ۱-۱: پرگزیدن لینوکس (منبع IDC 2002)

همانطورکه اشاره شد، چون لینوکس را می‌توان رایگان بدست آورد، سود ناشی از این سامانه-عامل عملاً بی معنی است. اما داشتن تقریبی از تعداد نسخه‌های آن، شاخص خوبی در زمینه موارد مرتبط با درآمد محصول، مانند کارخواهها^۱، نرمافزارها و موارد دیگر خواهد بود. نرخ سرعت رشد لینوکس و متوقف نشدن رشد آن در آینده نزدیک نیز از روی نمودار مشخص است. توجه کنید که نسخه‌های رایگان و خریداری شده هر دو باید در ارزیابی لحاظ شوند. به طور منطقی، از این نمودار می‌توان این‌گونه برداشت کرد که شرکتهای مختلف، سرمایه‌گذاری محصولات آینده خود را در بخش فنی را به سمت لینوکس پیش می‌برند تا از مزایای این رشد بهره‌مند شوند.

۱-۲- نگاهی گذرا به لینوکس و واژگان متن باز

وقتی شما خواندن این کتاب را به اتمام برسانید، درک بهتری از لینوکس، متن باز و نحوه به کارگیری آنها در کسب و کار خود پیدا خواهید کرد. در هر صورت فهم سطحی یک سری مفاهیم، به درک اولیه شما کمک خواهد کرد و همان‌طور که با کتاب پیش می‌رویم، این مفاهیم با جزئیات بیشتر بیان خواهند شد:

- **هسته^۲**: هسته، بخش اصلی سامانه‌عامل را تشکیل می‌دهد که کار آن مدیریت منابع اصلی (مثل حافظه، فرآیندها^۳ و سایر موارد سامانه است.
- **گنو^۱**: این لغت از کنار هم قراردادن حروف اول GNU'S NOT UNIX (گنو یونیکس نیست) شکل گرفته است. چون لینوکس به خودی خود هسته سامانه‌عامل است،

¹ Clients

² Kernel

³ Process

سامانه گنو بخشهای دیگری را (مانند مترجمها^۱، ویرایشگرها و سایر موارد) که مورد نیاز یک سامانه کامل هستند، ارائه می‌دهد و لینوکس با استفاده از این ابزارها تبدیل به یک سامانه‌عامل می‌شود. به همین دلیل است که لینوکس را معمولاً گنو/لینوکس می‌نامند و این مسأله مهم همواره باید در نظر گرفته شود.

- **توزیع^۲:** شرکتهای مختلفی که برنامه‌های کاربردی برای لینوکس می‌نویسند، اغلب با استفاده از توزیعهای رایج لینوکس این کار را انجام می‌دهند. برخی از رایج‌ترین نسخه‌های لینوکس عبارتند از: Red Flag Connectiva, Debian, SuSE, RedHat, Mandrake. عرضه‌کنندگان توزیعها، هسته لینوکس را با بخش‌های مختلفی از سامانه گنو و برنامه‌های متن‌باز دیگر ترکیب می‌کنند و پس از اعمال اصلاحات مورد نظر خود، مجموعه حاصل را به صورت یک سامانه‌عامل کامل و آزمون شده عرضه می‌کنند. این توزیعهای مختلف همگی لینوکس هستند، ولی هر کدام برای یک یا چند امر خاص طراحی شده‌اند و قابلیت خاص خودشان را به کاربران ارائه می‌دهند.
- **بسته نرم‌افزاری^۳:** هر کدام از برنامه‌هایی که روی سامانه‌عامل لینوکس نصب می‌شوند، به صورت بسته‌های نرم‌افزاری نیز توزیع می‌شوند. عموماً یک بسته نرم‌افزاری به مجموعه‌ای از پروندهای که برای انجام یک کار خاص در کنار هم قرار می‌گیرند، اطلاق می‌شود. چون بسته‌های نرم‌افزاری مختلف اغلب از یک سری پرونده به‌طور مشترک استفاده می‌کنند، این وابستگیها با کمک ابزارهای خاص نصب کنترل و مدیریت می‌شوند.
- **نرم‌افزارهای آزاد^۴:** مفهوم نرم‌افزار آزاد که با تاسیس "بنیاد نرم‌افزار آزاد"^۵ شکل گرفت به این معناست که نرم‌افزارها باید در اختیار دیگران قرار داده شوند و هر کس باید دسترسی یکسانی به کد منبع یا طراحی سامانه داشته باشد. [هر کس می‌تواند آن را به هر منظوری اجرا کند، به دیگران کپی آن را بدون محدودیت بدهد، یا که منبع آن را بررسی و تغییر دهد.]
- **متن‌باز^۶:** این واژه عموماً در دو زمینه به کار می‌رود: یکی به عنوان یک اصطلاح بازاریابی برای نرم‌افزار آزاد و مورد دیگر به عنوان یک نوع متداول‌تری توسعه که

¹ GNU

² Compiler

³ Distribution

⁴ Software package

⁵ Free software

⁶ Free Software Foundation (FSF)

⁷ Open Source

جزئیات آن در بخش سوم بررسی خواهد شد. از آنجایی که پیشگامان حرکت نرمافزارهای مجانی مخالف استفاده از اصطلاح آزاد هستند، این واژه برای از بین بردن تصور غلط نرمافزار رایگان به جای نرمافزار آزاد ساخته شد. متن باز یک مجوز نیست، بلکه مجموعه شرایطی را تعیین می‌کند که مجوزها برای متن باز بودن، باید لحاظ کنند.

- **انجمن^۱:** به هر مجموعه‌ای از توسعه‌دهندگان نرمافزار (برنامه‌نویسان) که با همکاری یکدیگر بر روی یک پروژه نرمافزاری کار می‌کنند، انجمن گفته می‌شود. این افراد می‌توانند دانش‌آموزان، افراد با علاقه‌خاص، برنامه‌نویسان در استخدام شرکتها، رقیبان و یا حتی مشتریان باشند.
- **نگهدارنده^۲:** یک فرد، کمیته و یا مجموعه‌ای از مدیران که در مورد قبول یا رد تغییرات در کد پروژه متن باز تصمیم می‌گیرند. نگهدارنده را می‌توان مدیر پروژه تلقی کرد. معروف‌ترین نگهدارنده در دنیای متن باز، لینوس توروالدز - طراح هسته لینوکس - می‌باشد.
- **GPL^۳:** این واژه مخفف مجوز کلی و عمومی گنو می‌باشد. GPL به طور مفصل در فصل سوم شرح داده می‌شود. تا رسیدن به آن فصل دانستن این که GPL، پرکاربردترین مجوز متن باز است، کافی است. GPL مجوزی است که هسته لینوکس [و بسیاری از نرمافزارهای آزاد و متن باز دیگر] را تحت پوشش خود دارد و مقرر می‌کند که هر تغییر در کد منبع باید به انجمن برگردانده شود. [یک نرمافزار با مجوزهای GPL همیشه GPL است و حق معنوی مشارکت‌کنندگان در توسعه آن همیشه محفوظ باقی می‌ماند.]

فهم این واژگان برای شروع کار کافی است. همان طور که کتاب را می‌خوانید به عقب برگردید و مطمئن شوید که واژه‌ها را خوب فهمیده‌اید.

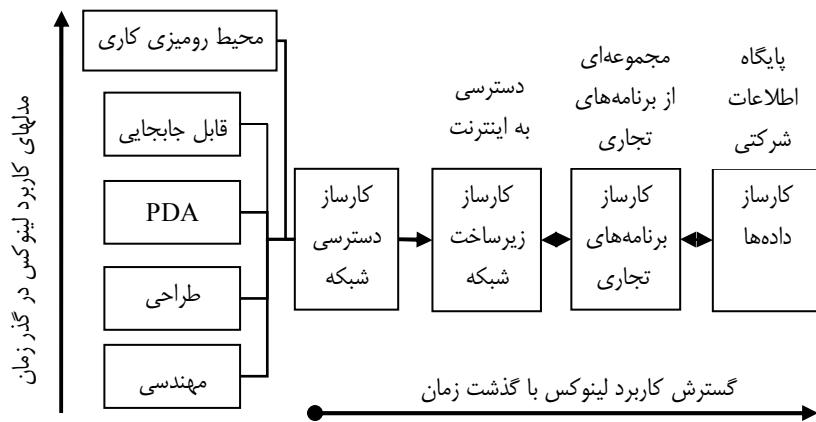
۱-۳- حجم کاری لینوکس

شکل ۱-۲ موارد کاربرد لینوکس و نحوه گسترش آن را در گذر زمان نشان می‌دهد. لینوکس در سمت کارساز قویترین ابزار در "لبه شبکه" (نقطه دسترسی به شبکه) است که این، عمدتاً به واسطه ترکیب پرقدرت سامانه‌عامل لینوکس و کارساز وب آپاچی محقق شده است. کاربردهای لینوکس همراه با رشد آن به تدریج به سمت کاربردهای زیربنایی گسترش می‌یابد.

¹ Community

² Maintainer

³ GNU General Public License



شکل ۱-۲: رشد حجم کار لینوکس با گذشت زمان

موضوعی که در اینجا مطرح می‌شود این است که این تحول چقدر طول می‌کشد. در سمت کارخواه، لینوکس با داشتن سابقه زیاد به عنوان یک سامانه‌عامل شبیه به یونیکس، طبیعتاً به واسطه ایستگاه‌های کاری مهندسی و سامانه‌های توسعه برنامه‌های کاربردی شهرت پیدا کرده است. لینوکس همچنین در عرصه وسایل توکار^۱ نیز نفوذ کرده است.

سؤالی که مطرح می‌شود این است که آیا لینوکس جای خود را روی میزهای کار اداری به عنوان یک سامانه‌عامل پایدار باز خواهد کرد یا خیر؟

۱-۴-۱- مزایای تجاری

لينوكس به سرعت از یک وسیله سرگرمی به یک محیط مطمئن و قابل استفاده در بسیاری از کسب و کارها تغییر نقش داده است. یکی از مهمترین دلایل گسترش آن در دنیای تجارت، توسعه چشمگیر اینترنت بود. کارسازهای اینترنت، لینوکس را راه حل مناسبی برای به کارگیری سامانه‌های عمومی شده یافتدند. مجوز سامانه‌عامل یا برنامه‌های اصلی کاربردی متن باز مانند کارساز وب آپاچی و یا برنامه‌های دیگر مورد نیاز اینترنت که تحت لینوکس اجرا می‌شوند، به گونه‌ای است که هزینه‌ای برای آنها دربرداشت. هم اکنون محیطهای تجاری لینوکس را به عنوان یک جایگزین قابل اطمینان نسبت به محیطهای دیگر از لحاظ هزینه، منابع و رویکردهای کنترلی می‌دانند. در ادامه برخی از مزایای اقتصادی لینوکس آورده شده است.

۱-۴-۱-۱- هزینه

این واقعیت که لینوکس می‌تواند بدون پرداخت حق الامتیاز به طور رایگان عرضه شود، یکی از مزایای تجاری لینوکس به شمار می‌رود. به هر حال باید توجه داشت که اگر چه توسعه و به کارگیری لینوکس بدون پرداخت هزینه مجاز ممکن است، این بدان معنی نیست که لینوکس

¹ Embedded

و یا هر سامانه‌عامل دیگر از لحاظ هزینه‌های مربوط به نصب، نگهداری، پشتیبانی و آموزش نیز رایگان می‌باشد. در فصل ششم، ما نگاه مفصلی به هزینه‌های جانبی لینوکس و نرم‌افزارهای متن‌باز دیگر خواهیم داشت.

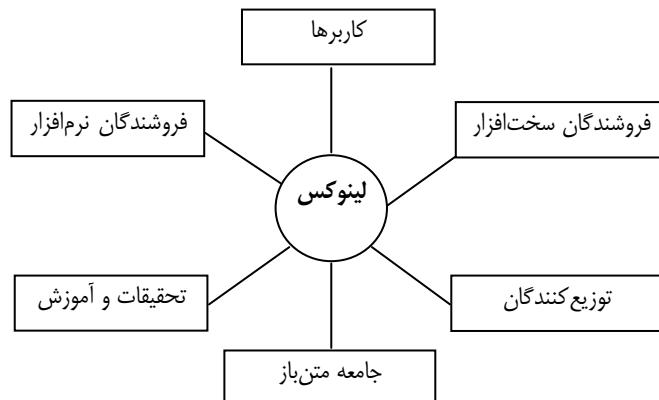
۱-۴-۲- در دسترس بودن مطالب آموزشی

لینوکس بیش از ده سال قدمت دارد، بنابراین منابع آموزشی بسیاری در این زمینه موجود است. هزینه‌های کم لینوکس، بسیاری از مؤسسه‌های تحقیق و پژوهش مثل دانشگاه واترلو در آنتاریوی کانادا، دانشگاه ایلینویز در اوربانا شامپین، مرکز ملی کاربردهای ابررایانه‌ای، NCSA و بسیاری از مؤسسه‌های دیگر را به فکر استفاده از لینوکس انداخته است. این رویدادها در طول چند سال رخ داده است. مثلاً گروه ESIEE که یکی از مراکز آموزش علمی و فنی پیشرفته فرانسه است، به مدت سه سال است که از لینوکس استفاده می‌کند. زیرا لینوکس متن‌باز است. انگیزه آنها از این کار همان گونه که خودشان اظهار داشته‌اند، بسیار ساده است: گروه آنها برای جامعه توسعه می‌یابد و جامعه برای گروه آنها توسعه می‌یابد. این کار یک نوع برگشت سرمایه برای آنهاست. در استرالیا، دانشگاهی^۱ تمام آزمایشگاه‌ها یونیکس خود را ظرف مدت یک سال به لینوکس تبدیل کرد و در نتیجه فارغ‌التحصیلان جدید به‌طور کامل تحت سامانه‌عامل لینوکس آموزش دیدند.

نسل جدید برنامه‌نویسان نیز درحال کسب تجربه تحت لینوکس و واسطه‌ای برنامه‌نویسی کاربردی (API) آن هستند. این نسل جدید با برنامه‌های کاربردی متن‌باز مشهوری مثل کارساز وب آپاچی نیز آشنا است.

شكل ۳-۱ گروه‌های اصلی دخیل در جنبش لینوکس را نشان می‌دهد. با رشد و توسعه سازمان و نیاز به داشتن نیروی کار مستعد برای راهبری زیرساخت شبکه، کارسازهای برنامه‌های کاربردی و مراکز داده افراد مستعد در زمینه لینوکس یکی از در دسترس‌ترین منابع مورد نیاز شما خواهد بود.

¹ New south wales



شکل ۱-۳: افراد سهیم در لینوکس

۱-۴-۳-۳- پشتیبانی

یکی از بحثهای داغ در مورد لینوکس این است که پشتیبانی از طریق چه مرجعی صورت گیرد. یک طرف این بحث بر آن است که چون لینوکس توسط جمعی از توسعه‌دهندگان - که ارتباط کمی با هم دارند - نگهداری می‌شود، خصمانت و پشتیبانی آن مورد شک است. طرف دیگر عقیده دارند چون کد منبع در دسترس همه است، هر کسی می‌تواند به نوعی پشتیبانی را فراهم کند و این پشتیبانی، هزینه کمتر و اطمینان بیشتری به همراه دارد. اینکه شما کدام طرف بحث را بپذیرید بستگی به نوع سازمان فناوری اطلاعات و تجربه قبلی شما در تعامل با عرضه‌کنندگان مختلف خدمات پشتیبانی دارد. به هر حال شما در انتخاب نوع پشتیبانی، قدرت انتخاب دارید. فصل هشتم به جزئیات راه حل‌های پیش رو برای پشتیبانی و مزایای هریک خواهد پرداخت.

۱-۴-۴- توسعه نرم‌افزار

لينوكس برای تیم برنامه‌نویسی Fluent اصلی‌ترین بستر محسوب می‌شود. Fluent یکی از شرکتهای پیشرو در ارائه نرم‌افزارهای مهندسی به کمک رایانه^۱ است. لینوکس، بستری پایدار و قدرتمندی با گنجینه‌ای از ابزارهای رایگان را در اختیار برنامه‌نویسان این تیم قرار داده است. علاوه بر این، برنامه‌نویسان از طریق شبیه‌ساز VMware که تحت لینوکس اجرا می‌شود، می‌توانند به برنامه‌های کاربردی ویندوز دسترسی داشته باشند. VMware در واقع برنامه‌ای است (در نسخه‌های تحت ویندوز و لینوکس) که یک پردازنده اینتل را شبیه‌سازی می‌کند و به

¹ Computer Aided Engineering

برنامه‌نویسان امکان دسترسی همزمان به سامانه‌های عامل دیگر مثل، ویندوز و یونیکس را در یک ماشین می‌دهد. آمار نشان می‌دهد که ویندوز با در اختیار داشتن ویژوال استودیو، گوی سبقت را در بازار توسعه نرم‌افزار ریوده است. با این حال اگر بازار لینوکس را تقسیم‌بندی کنیم، می‌بینیم که توسعه نرم‌افزار در حال حرکت به سمت لینوکس است که یکی از دلایل آن بازار آموزش و تحقیقات می‌باشد که قبل‌آن اشاره شد.

۱-۴-۵- ارتقاء

هر ساله شرکتها برای داشتن خروجی بهتر میلیونها دلار صرف ارتقاء برنامه‌های کاربردی یا محیط‌های برنامه‌نویسی می‌کنند. در بسیاری از این موارد انگیزه ارتقاء به خاطر نیاز به مجوز و سخت‌افزاری است که از طرف عرضه‌کنندگان اعمال می‌شود. با لینوکس همه چیز کنترل شده است. شما می‌توانید هزینه پشتیبانی و ارتقاء را محاسبه کرده و سپس در مورد اعمال تغییرات تصمیم بگیرید. در حالت کلی کاربران به ارتقاء سامانه راضی نمی‌شوند، مگر اینکه در نهایت منجر به سود تجاری بیشتر شود و سرمایه صرف شده برای ارتقاء به همراه خطرات آن، درآمدی داشته باشد. یکی از زیبایی‌های متن‌باز این است که تصمیم‌گیرنده شما هستید، نه عرضه‌کننده. سخت‌افزار نیز به تبع ارتقاء نرم‌افزار، ارتقاء خواهد یافت. در جامعه متن‌باز، برنامه‌نویسان زیادی از سامانه‌هایی استفاده کرده‌اند که از عمر آنها بیش از ۱۰ سال می‌گذرد و از کار افتاده محسوب می‌شوند.

شرکتها به خوبی می‌دانند که هر تغییری، خطرات و فرصت‌های دربردارد که مانند هر فعالیت دیگری در سازمان، به اندازه‌گیری و کیفیت‌سنجی نیاز دارد. اگر شما قصد به کارگیری محیط کاری لینوکس را داشته باشید، هیچ مجوزی در دنیای لینوکس و متن‌باز شما را مجبور به تغییر، افول یا ارتقاء آن محیط نخواهد کرد. شما کنترل کاملی بر محیط خود دارید. بنابراین موفقیت یا شکست شبکه کاری و در نهایت کسب و کار خود را خودتان کنترل می‌کنید.

۱-۵- موانع رشد لینوکس

درست است که لینوکس یک رویکرد مناسب برای صرف هزینه کمتر، بالا بردن کارایی و شریک شدن با جامعه بزرگی از برنامه‌نویسان می‌باشد، اما لینوکس اکسیری نیست که پاسخگوی همه کاستیهای فناوری اطلاعات باشد. در زیر برخی از موانعی که بر سر راه لینوکس قرار دارد، آورده شده است. در عین حال تلاشهای بسیاری برای از بین بردن موانع رشد لینوکس در دنیای فناوری‌های پیشرفته می‌شود که زمان، بسیاری از این موانع را از سر راه برخواهد داشت.

۱-۵-۱- دردسترس بودن برنامه‌های کاربردی

لينوکس بسياری از موفقیتهای خود را مرhone کارساز وب آپاچی است. طبق تحقیقات صورت گرفته توسط Netcraft در اوخر سال ۲۰۰۱ کارساز وب توسط بیش از پنجاه درصد از سایتهاي شبکه استفاده می‌شده است [آمار سال ۲۰۰۵ حدود ۷۷٪ است]. ترکیب لینوکس و آپاچی، یک محیط مقرر برای صرفه را با کارایی بالا- که به سادگی قابل گسترش است- برای کارسازها و مؤسسات به وجود می‌آورد.

علاوه بر آپاچی، برنامه‌های کاربردی زیادی توسط انجمنهای آموزشی و تحقیقاتی و یا شرکتهای مختلف توسعه پیدا کرده‌اند.

مسئله کلاسیک مرغ و تخم مرغ برای محیطهای زیادی صادق است و لینوکس از این قاعده مستثنی نیست. در حالی که شرکتهای زیادی منتظر برنامه‌های کاربردی قبل از توسعه لینوکس هستند، تولیدکنندگان نرم‌افزار منتظر مشتریان هستند تا لینوکس را به کارگیرنده، سپس برنامه‌های آن را تولید کنند.

۱-۵-۲- خطر^۱ تجاری

لینوکس و اغلب نرم‌افزارهای متن‌باز خطرات تجاری خاص خود را به همراه دارند. پیاده‌سازی هر فناوری جدیدی در سازمان با خطرهایی همراه است که باید مدیریت شوند. لینوکس و متن‌باز نیز از این قاعده مستثنی نیستند. فرآیندهای تجاری که از قبل موجود نیستند باید شکل گیرند. شرکتهای بسیاری درک کرده‌اند که لینوکس از عهده مدیریت این خطرات و افزودن بر سرمایه اقتصادی به خوبی برمی‌آید. بخش سوم این کتاب به بررسی این خطرات و نحوه مدیریت آنها می‌پردازد. در واقع متن‌باز فرصت‌های جدیدی برای افزایش کارایی و تولید محصول و جلب رضایت مشتری به وجود می‌آورد.

۱-۶- نقش آفرینان متن‌باز

جامعه متن‌باز متشکل از افرادی است که در کل جهان روی پروژه‌های متن‌باز کار می‌کنند. بعضی از این افراد این کار را به عنوان یک وظیفه و برخی نیز صرفاً از روی علاقه انجام می‌دهند. بعضی به علت اعتقاد به متن‌باز و بعضی برای پیدا کردن جواب مسائل خود به این کار می‌پردازند. تعدادی از این برنامه‌نویسان عضو انجمنهای آموزشی و تحقیقاتی هستند. جدول ۱-۴ نمودار زمانی رویدادهای مهم و اصلی را که از زمان حیات لینوکس و متن‌باز رخ داده است - نشان می‌دهد.

در بخش سوم این کتاب ما دلایل تشویق کارمندان توسط مدیران در روی آوردن به پروژه‌های متن‌باز را شرح خواهیم داد. تاکنون افراد برجسته و رویدادهای بارزی که ارزش شناخته‌شدن داشته باشند و فهمیدن تأثیر آنها در جامعه متن‌باز مفید باشد، خیلی زیاد نبوده است. در عین حال آوردن نام همهٔ کسانی که به نحی در این کار نقش داشته‌اند نیز ممکن نیست. اما باید این نکته را در نظر داشت، افرادی هستند که نقش تعیین‌کننده‌ای در این کار داشته‌اند، ولی نام آنها ذکر نشده است. به ترتیج متوجه خواهید شد که متن‌باز در واقع یک بازی تأثیرگذاری و ایجاد روابط است. به محض اینکه شما جوامع مفید برای کسب و کارتان را شناختید، به ارزش صرف وقت و انرژی برای برقراری ارتباط با رهبران و اعضاء آن جوامع پی خواهید برد.

^۱ Risk

بنیاد نرم افزار آزاد (FSF) [توسط ریچارد استالمان (RMS)] تشکیل شد.	۱۹۸۴
اولین نسخه GPL را عرضه کرد.	۱۹۸۹
دومین نسخه GPL را عرضه کرد.	ژوئن ۱۹۹۱
اولین کد هسته لینوکس عرضه شد.	سپتامبر ۱۹۹۱
لینوکس ۱.۰ عرضه شد.	مارس ۱۹۹۴
شرکت RedHat تأسیس شد.	۱۹۹۴
گروهی از برنامه نویسان دور هم جمع شدند تا بر روی کارساز وب آپاچی کار کنند.	۱۹۹۵
لینوکس نسخه ۲ هسته عرضه شد.	ژوئن ۱۹۹۶
پروژه میز کار KDE آغاز شد.	اکتبر ۱۹۹۶
مقاله "کاتدرال و بازار" منتشر شد.	۱۹۹۷ مه
پروژه میز کار GNOME آغاز شد.	۱۹۹۷
واژه متن باز رسمی شد.	فوریه ۱۹۹۸
نسخه ۲.۴ هسته لینوکس عرضه شد.	ژانویه ۲۰۰۱

شکل ۱-۴ نمودار زمانی وقایع لینوکس و متن باز

لینوکس توروالدز^۱: وی احتمالاً مشهورترین شخصیت در جهان لینوکس و متن باز است. توروالدز طراح و نگهدارنده هسته لینوکس است. او اجزای مستقل از بستر (سخت افزار) هسته را نگهداری می کند و اجزای هسته مرکزی هسته را -که مختص معماری اینتل هستند- نگهداری می کند. واسطه های لینوکس برای معماری های دیگر نیز به وسیله اشخاص یا گروه های دیگری به صورت جداگانه نگهداری می شود.

آلن کوکس^۲: آلن تا پاییز سال ۲۰۰۱ جانشین اول لینوکس محسوب می شد. در فصل دوم خواهید دید که لینوکس، هسته لینوکس را -که هم اکنون در حال توسعه است- نگهداری می کرد. در حالی که آلن هسته عرضه شده را نگهداری می نمود. طبق قاعده جامعه متن باز، [در خصوص نگهدارنده نرم افزار متن باز] هر گاه آلن اراده می کرد، می توانست روى موضوع دیگری کار کند. او ابتدا یک جانشین برای خود انتخاب کرد. اگر چه مدت زیادی نیست که آلن به این کار مشغول است، ولی نفوذ قابل توجهی در جامعه دارد و از اعتبار و احترام فراوانی برخوردار است.

مارچلو توساتی^۳: لینوز و آلن وظیفه نگهداری هسته پایدار لینوکس (هسته در حال تولید) را به مارچلو سپردند. وی یکی از توسعه دهندگان هسته لینوکس است که لیاقت بالایی در کار با جامعه

¹ Linus Torvalds

² Alan Cox

³ Marcelo Tosatti

برنامه‌نویسان از خود نشان داده است. [مارچلو یک جوان ۲۰ ساله بزریلی است که از ۱۳ سالگی با لینوکس کار می‌کند]

دیو میلر^۱: میلر مسئول اعمال تغییرات صورت گرفته در پشت‌های شبکه در هسته لینوکس است و در مورد یا رد این تغییرات تصمیم می‌گیرد. میلر را می‌توان نگهدارنده بخش شبکه هسته لینوکس دانست.

ریچارد استالمان^۲: مؤسس FSF و طراح مجوز GNU GPL می‌باشد. ریچارد همچنین نقش تعیین‌کننده‌ای در جنبش نرم‌افزارهای آزاد دارد. افراد زیادی از نظرات و راهنماییهای او در قسمتهای مختلف GPL بهره می‌گیرند.

بروس پرنز^۳: مدیر پروژه دیبان - که در فصلهای بعدی به تفصیل شرح داده خواهد شد - و تدوین‌کننده یک قرارداد اجتماعی برای جا انداختن مجموعه اصول در کار با نرم‌افزارهای آزاد می‌باشد. وقتی واژه "متن باز" شکل گرفت، این قرارداد، "تعریف متن باز" (OSD) نام گرفت. به همین دلیل بروس را به عنوان مؤلف اصلی آن می‌شناسند. فصل سوم در این باره توضیحات بیشتری می‌دهد.

اریک ریموند^۴: او به عنوان انسان‌شناس و رهبر فکری جامعه متن باز شناخته شده است. شهرت اریک به خاطر مقاله‌اش با عنوان "کاترال^۵ و بازار" است. این مقاله فرآیند توسعه متن باز، علت پیشرفت آن و نحوه پایداری و استمرار آن را شرح می‌دهد. بخش سوم این کتاب درباره این موارد بحث می‌کند.

لری آگوستین^۶: لری مدیر اجرایی شرکت نرم‌افزاری VA است. شرکت نرم‌افزاری VA در ابتدا با نام سامانه‌های لینوکس VA شکل گرفت. لری از اولین افرادی بود که به اهمیت لینوکس و متن باز پی برد. او اولین شرکت سخت‌افزاری را که از لینوکس پشتیبانی می‌کرد ساخت. زمانی که شرکتهایی مثل HP، IBM و Dell سخت‌افزارهایی برای پشتیبانی لینوکس عرضه می‌کردند، لری مدل تجاری شرکت خود را به فروش نرم‌افزارهایی که فرآیند توسعه متن باز را در شرکتها پشتیبانی می‌کردند، تغییر داد.

تیم اوریلی^۷: او به عنوان رئیس انتشارات O'reilly جایگاه خاصی در جنبش متن باز دارد. شبکه کتابهای O'reilly فناوریهای اصلی متن باز مثل Perl و موارد دیگری را که کاربرد وسیعی در اینترنت دارند، پوشش می‌دهد. او با چاپ افکار و ایده‌های جامعه متن باز، نقش سخنگوی این جامعه را ایفا می‌کند.

¹ Dave Miller

² Richard Stallman

³ Bruce Perens

⁴ Eric Raymond

⁵. به کلیساها مرکزی کاترال گویند که اغلب از ساختمانی بزرگ با معماری منحصر بفرد تشکیل شده‌اند.

⁶ Larry Augustin

⁷ Tim O'reilly

برین بلندورف^۱: برایان یکی از بنیان‌گذاران مؤسسه نرمافزاری آپاچی است. از آنجایی که آپاچی و متن‌باز در ارتباط با هم هستند، او نه تنها تأثیر به سزایی بر رشد آپاچی داشته است، بلکه در جامعه متن‌باز نیز به شدت تأثیرگذار بوده است.

مایکل تیمان^۲: مایکل اولین شرکت بازرگانی مبتنی بر مدل تجاری متن‌باز را با عنوان Cygnus تأسیس کرد. Cygnus زیرساخت لازم برای یکی از اجزای کلیدی لینوکس، یعنی مترجمها را ارائه کرد. وی همچنین در تعریف مدل‌های تجاری مبتنی بر تأمین پشتیبانی و ارائه خدمات برای نرمافزارهای آزاد، پیشرو بود. Cygnus در سال ۲۰۰۰ به RedHat پیوست.

باب یانگ^۳: اعتباری که باب امروزه کسب نموده است، ناشی از موفقیت محبوب‌ترین توزیع لینوکس یعنی RedHat می‌باشد. باب برای ایجاد یک محیط تجاری قوی با استفاده از سامانه-عامل تجاری لینوکس، از استعداد بازاریابی خود استفاده کرد. امروزه او یکی از رهبران اصلی شرکت RedHat است و در اغلب رویدادهای لینوکس و متن‌باز سخنرانی می‌کند.

جرمی آلیسون^۴: جرمی یکی از مدیران پروژه سامبا^۵ است. همان‌گونه که خواهید دید، سامبا یکی از نرمافزار کلیدی برای برقراری ارتباط بین جهان ویندوز و یونیکس/لینوکس است. سامبا یکی از فناوریهایی است که به همراه اکثر توزیعهای لینوکس عرضه می‌شود.

میگوئل دو ایکازا^۶: میگوئل مؤسس و مدیر ارشد فناوری شرکت Ximian است. او سالها مدیریت پروژه توسعه محیط رومیزی گنو را بر عهده داشت و اکنون پروژه متن‌باز دیگری با نام Mono را رهبری می‌کند. این پروژه، پیاده‌سازی زیرمجموعه‌ای از محیط کاری دات نت (.Net).^۷ مایکروسافت، به صورت متن‌باز است.

¹ Brian Behlendorf

² Michael Tiemann

³ Bob Young

⁴ Jeremy Allison

⁵ SAMBA

⁶ Miguel de Icaza

.۷ Ximian در سال ۲۰۰۴ به شرکت Novell ملحق شد و Mono نیز تحت نظرارت این شرکت هدایت می‌شود.

۷-۱- جمع‌بندی

اکنون با خواندن این مقدمه، شما اطلاعات کافی را برای مطالعه قسمتهای بعدی کتاب دارید. در ادامه خواهید دید که لینوکس فرصت‌های بسیاری برای کاهش هزینه‌های فناوری اطلاعات در اختیار شما می‌گذارد. در عین حال نباید مدیریت خطرهای تجاری ناشی از آن را نادیده گرفت. با فهم واژگان اصلی و شناخت نقش آفرینان این حوزه، می‌توان به بررسی نحوه بهره‌مندی از مزایای متن‌باز و مدیریت خطر پرداخت.

فصلهای بعدی بخش اول، دانش شما را در مورد سامانه‌عامل لینوکس گسترش می‌دهند و جزئیات بیشتری از متن‌باز و مجوز مربوطه را روشن می‌کنند. این بخش با ارائه یک دید کلی نسبت به جوامع متن‌بازی که احتمال مواجهه و کار با آنها بیشتر است، به پایان می‌رسد. در بخش دوم نگاهی به جنبه عملی لینوکس می‌اندازیم. این بخش شما را تشویق به بررسی نیازهای شرکت می‌کند و اطلاعاتی درمورد پشتیبانی، نگهداری و هزینه‌های لینوکس در اختیارتان قرار می‌دهد. بخش سوم به بررسی مدیریت خطرها و فرصت‌های متن‌باز می‌پردازد. این بخش علاوه بر شرح نحوه کار با پروژه‌های متن‌باز، نحوه پیاده‌سازی متالوژی توسعه آن در محیط کار را نیز شرح می‌دهد.

فصل دوم

لینوکس – قلب سامانه‌عامل

معمولًاً منظور از "لینوکس" کل سامانه‌عامل است و در اکثر موارد چنین تلقی از آن اشکالی ندارد. هر چند که لینوکس در واقع بخش کوچکی از یک سامانه‌عامل کاری را تشکیل می‌دهد. از نظر فنی، لینوکس فقط هسته است؛ بخشی که باعث می‌شود سامانه به درستی به وظایف خود را عمل کند. در کنار هسته اجزای دیگری وجود دارند که یک سامانه‌عامل را کامل می‌کنند. ابزارهایی که به عنوان بخشی از سامانه‌عامل لینوکس ارائه می‌شوند و در کنار هم را تبدیل به یک سامانه‌عامل کامل می‌کنند، سامانه گنو نامیده می‌شوند.

از این رو است که ریچارد استالمون ادعا می‌کند نام لینوکس برای سامانه‌عامل اشتباه است، بلکه باید کل سامانه را گنو/لینوکس نامید. مؤلفه‌های دیگری مثل سامانه پنجره‌ای X11 واسطه‌های میزکار، ابزارهای به اشتراک گذاردن پروندها و سایر ابزارها نیز باید به لینوکس اضافه شوند که با این حساب نام گنو/لینوکس هم کاملاً دقیق نیست. از نظر قانونی، لینوکس علامت تجاری ثبت شده برای "نرم‌افزار سامانه‌عامل برای تسهیل عملیات رایانه‌ای" می‌باشد. لینوس توروالدز مالک این علامت تجاری است.

تصمیم به توسعه لینوکس در سازمان، نیازمند درک درست مؤلفه‌های لینوکس است. بنابراین اهداف این فصل درک مطالب زیر می‌باشد:

- یک دید ساده از سامانه‌عامل
- هسته لینوکس و تفاوت‌های آن با انواع دیگر هسته‌ها
- چگونگی توسعه و بهروزرسانی هسته لینوکس و سامانه نسخه‌بندی آن
- نحوه گسترش کارایی هسته با به کارگیری پیمانه‌ها و وصله‌ها^۱
- ملاحظات جداسازی^۲ توسط لینوکس.
- جنبه‌های تعدد بسترها در لینوکس
- کاربردهای مهم لینوکس

این فصل چندین مفهوم فنی پیچیده را مورد بررسی قرار می‌دهد. طبیعت هسته‌های سامانه‌عامل به گونه‌ای است که بحث درباره آن را پیچیده می‌کند. هدف، آشنایی با این مباحث پیشرفته برای درک نحوه تأثیر آنها بر تصمیم‌گیری در زمینه به کارگیری، مهاجرت (تغییر بستر یا سامانه‌عامل) و انجام پروژه‌ها است. همچنین اغلب مدیران فناوری اطلاعات نگران تکرار جنگهای کهن یونیکسی هستند که باعث ناهنجاریهای در بازار شدند. افزایش درک شما از لینوکس و ساختار آن، در تصمیم‌گیریهای فناوری اطلاعات به شما کمک نموده و از نگرانیهای شما می‌کاهد. خود را برای پاسخ به این سؤال که "چرا لینوکس متفاوت است آماده کنید؟"

¹ Patches

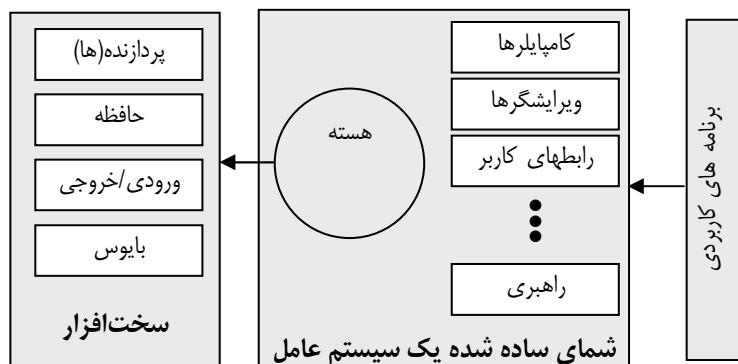
² Fragmentation

در فصل پنجم به بررسی توزیعهای لینوکس خواهیم پرداخت. توزیعها ترکیبی هستند از هسته لینوکس، سامانه گنو و اجزای مورد نیاز برای داشتن سامانه‌ای کارآ. در برخی موارد یک توزیع، کارآیی مورد نیاز برای یک سامانه کامل را در بردارد و در برخی موارد نیز نیاز به تهیه برنامه‌های کاربردی اضافه نی از است.

۲-۱- سامانه عامل

در زمینه آنچه که سامانه عامل باید دربرداشته باشد، استاندارد صدرصد پذیرفته شده‌ای وجود ندارد، زیرا اجزایی که در هر سامانه عرضه می‌شوند، در طول زمان به طور چشمگیری رشد و تغییر کرده است. در اواسط دهه ۸۰ میلادی- وقتی که بحث شبکه‌های رایانه‌ای هنوز نوپا بود - شرکتهای زیادی قراردادهای شبکه خود (مثل TCP/IP) را به همراه سامانه‌های عامل محبوب می‌فروختند. با تکامل شبکه‌های رایانه‌ای و تبدیل آنها به یکی از نیازهای اصلی برنامه‌های کاربردی، قراردادهای شبکه جزء اصلی اکثر سامانه‌های عامل شدند.

بحث اجزای تشکیل‌دهنده سامانه عامل به جایی رسیده است که بعضی عرضه‌کنندگان اجزای خاصی را ارائه می‌دهند و برخی دیگر نمی‌دهند. به هر حال، اغلب عرضه‌کنندگان، شکل ۲-۱ را به عنوان یک نمای ساده شده از مؤلفه‌های سامانه عامل مد نظر قرار داده‌اند. سامانه عامل لینوکس همه این اجزا را شامل می‌شود.



شکل ۱-۲: سامانه عامل

همان‌طور که می‌بینید، سامانه عامل، پل ارتباطی بین برنامه‌های کاربردی (که شما برای انجام کار خود به آنها نیاز دارید) و منابع سامانه (مثل بخش‌های مختلف سخت‌افزاری) است. برای داشتن برنامه‌های کاربردی بیشتر و بالا بردن بهره‌وری سامانه، سامانه عامل باید توانایی انجام کار بیشتر را داشته باشد. سامانه‌های عامل مصرف‌کنندگان، برنامه‌های کاربردی بسیاری (مثل پردازش تصویر، اجرای موسیقی و غیره) را به همراه دارند. سامانه‌های عاملی که برای دنیای شرکتی طراحی شده‌اند، بر برنامه‌های کاربردی کرده‌اند که قابلیت مدیریت کاربران زیاد را داشته باشد و هدف عمده آنها از این کار کنترل منابع رایانه با حداقل هزینه است.

بهره‌گیری از منابع در لینوکس از سه جنبه قابل بررسی است:

۱. مدیریت، ۲. راهبری، ۳. کنترل.

از نقطه نظر سامانه‌عامل، مدیریت منابع دید جامعی از منابع موجود رایانه و نحوه توسعه و به کارگیری آنها ارائه می‌دهد. مثلاً اگر شما یک خط هوایپیمایی را مدیریت می‌کنید، هوایپیماتان را در مسیرهایی به کار می‌گیرید که درآمد بهینه مناسب با آن مسیر را به همراه داشته باشد. به طور مشابه در لینوکس مؤلفه‌های مدیریت منابع به شما در مدیریت مؤلفه‌های سخت‌افزاری سامانه به طور یکپارچه کمک خواهند کرد.

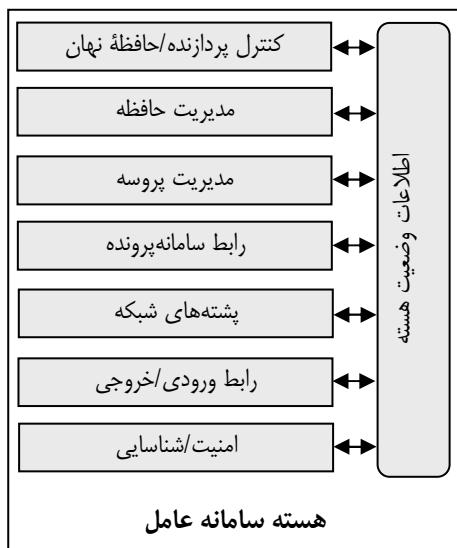
پس از توسعه منابع رایانه، ابزارهای سرپرستی منابع در لینوکس با کاراترین روش ممکن به پیکربندی منابع موجود کمک می‌کنند. در مورد مثال خط هوایپیمایی، شما از برپایی هوایپیما با مقدار سوخت مناسب، غذای خوب و مناسب برای مسافران و کارکنان مجرب اطمینان حاصل می‌کنید.

کنترل منابع در حوزه هسته سامانه‌عامل لینوکس قرار می‌گیرد. هسته منابع موجود را با توجه به طراحی و پیکربندی آنها کنترل می‌کند. درست مثل اینکه هوایپیما با ارتفاع و سرعت از پیش تعیین شده پرواز می‌کند.

۲-۲- هسته لینوکس

لينوکس از نظر فنی هسته سامانه‌عامل است. شکل ۲-۲ مؤلفه‌های اصلی هسته را نشان می‌دهد. برای درک بهتر، هسته را می‌توان به موتور هوایپیما شبیه کرد. موتور، توان مورد نیاز هوایپیما را تأمین می‌کند و این در حالی است که خلبان به طور مستقیم با موتور تعاملی ندارد. هسته، توان سامانه‌عامل است که کاربران رایانه هیچ‌گاه مستقیماً با آن سروکار ندارند. در واقع می‌توان گفت هسته نرم‌افزار عظیمی است که سخت افزار را کنترل می‌کند. هسته کنترل و اختیار منابع سامانه مثل دیسکها، حافظه، واحد پردازش مرکزی^۱ و غیره را در دست دارد. این وظیفه هسته است که همه درخواستهای ورودی (از طرف برنامه‌های کاربردی و کاربران) برای منابع سامانه را مدیریت کند و پاسخگو باشد. کارایی هسته در این زمینه به طراحی و پیکربندی آن بستگی دارد.

^۱ Central Processing Unit (CPU)



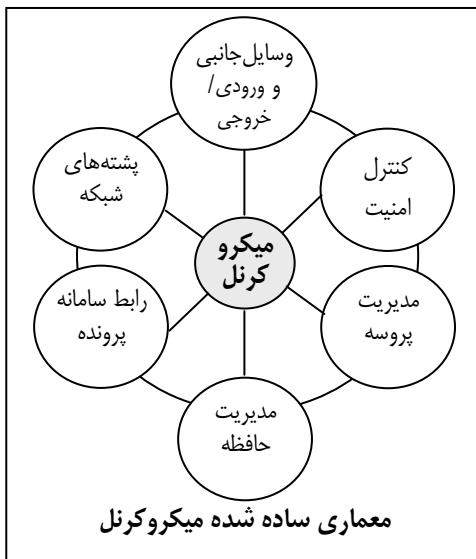
شکل ۲-۲: هسته و عملکردهای سیستم عامل یکپارچه

۲-۲-۱- طراحی هسته

لینوکس یک هسته یکپارچه^۱ است. همان طور که شکل ۲-۲ نشان می‌دهد، همه مؤلفه‌های هسته را می‌توان یک برنامه کاربردی خاص و عظیم که روی سامانه اجرا می‌شود، در نظر گرفت. این برنامه کاربردی ویژه دارای تواناییها و ویژگیهای خاص است که کار آن کنترل سامانه است. واژه "برنامه کاربردی" باید با برنامه‌های کاربردی - که برای انجام کارهای روزمره به کار می‌رود - اشتباه گرفته شود. هر اطلاعاتی در مورد حالت هسته، در اختیار همه اجزای هسته قرار می‌گیرد. نوع دیگری از هسته که در سامانه‌عامل پیشرفته به کار می‌رود، میکروهسته است. میکروهسته‌ها اکثر وظایف کنترلی اصلی را به چندین قسمت تقسیم می‌کنند. علاوه بر اینکه اطلاعات برای همه توابع قابل دسترسی است، پیامها از یک تابع به تابع دیگر منتقل می‌شوند.

شکل ۳-۲ یک معماری ساده میکروهسته را نشان می‌دهد.

^۱ Monolithic



شکل ۲-۳: معماری یک میکروکرنل نوعی

طرفداران میکروهسته‌ها بر این باورند که پیمانه‌ای^۱ بودن هسته، کارآیی و قابلیت انتقال آن را افزایش می‌دهد. قابلیت انتقال به این معنی است که کل سامانه‌عامل را بتوان از روی یک لوح فشرده اجرا و با خودتان جابه‌جا نموده و تنظیمات آن را منتقل کنید.

از سوی دیگر افرادی مثل لینوس توروالدز معتقدند که فرآیند انتقال پیام بین هر یک از پیمانه‌های کنترل، پیچیدگی را افزایش داده و هسته را مستعد کاستیها و نقاط ضعف میهم می‌کند. ریچارد استالمون هسته‌ای برای سامانه گنو طراحی کرد که مبتنی بر معماری میکروهسته بود و Hurd GNU نام داشت. ریچارد اعتراف می‌کند که پیچیدگی این هسته، فرآیند توسعه و رفع اشکال آن را به طور قابل توجهی کند کرد. این درحالی است که هسته یکپارچه توروالدز خیلی سریع برای پشتیبانی سامانه گنو قبل از آماده شدن Hurd GNU طراحی شد.

۲-۲-۲- پیمانه‌های هسته لینوکس

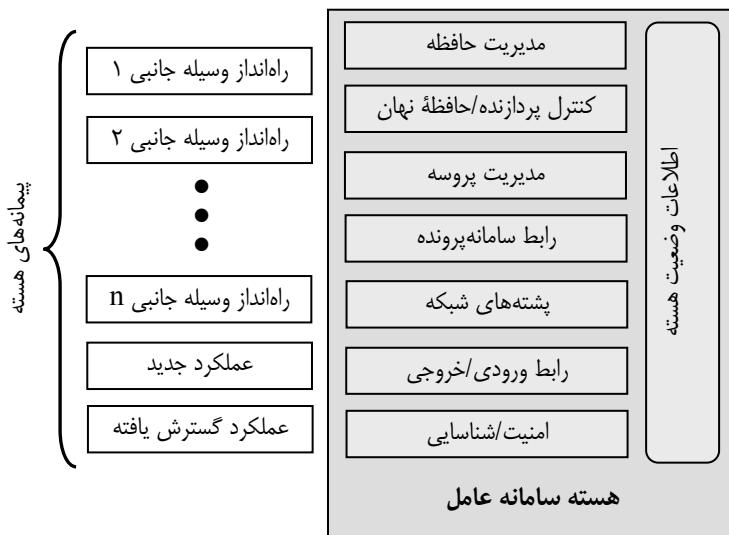
یکی از پیشرفتهای اصلی در نسخه ۲ هسته لینوکس، استفاده از پیمانه‌ها بود. قبل از این، هر کارآیی یا قابلیت جدیدی که به هسته اضافه می‌شد، باید در هسته ترجمه^۲ شده و سامانه برای استفاده از آن قابلیت جدید دوباره راهاندازی می‌شد. پیمانه‌ها مکانیزمی برای افزایش کارآیی

¹ Modular

² Compile

هسته لینوکس هنگامی که سامانه در حال کار است، فراهم می‌کنند. معمول‌ترین روش به کارگیری پیمانه‌ها از طریق راهاندازها (drivers) است. یک راهانداز، نرمافزاری است که بین سامانه‌عامل و وسیله سخت‌افزاری ارتباط برقرار می‌کند. در قیاس با هوایپما راهاندازها مثل کنترل جریان بنزین، راهبری هوایپما و مکانیزم‌های دیگری هستند که خلبان برای کنترل هوایپما به کار می‌گیرد. خلبانان بدون نگرانی از مشخصات موتور این کنترلها را به کار می‌گیرند. راهاندازها نیاز به داشتن اطلاعات در مورد وضعیت فعلی هسته دارند و همچنین باید از طریق اجزای هسته کنترل شوند.

اغلب راهاندازها توسط اشخاصی از انجمنهای مختلف توسعه متن‌باز نوشته می‌شوند. توجه داشته باشید که هنگام ساخت هسته، مشخص کردن همه دستگاه‌های مورد استفاده و پیکربندی‌های سامانه مورد نیاز کاربر نهایی، تقریباً غیر ممکن است. بدون پیمانه‌ها هر نرمافزار راهاندازی باید درون هسته ترجمه شود که این امر اندازه هسته را به طور چشمگیری افزایش می‌دهد و کارایی سامانه را پایین می‌آورد. شکل ۴-۲ پیمانه‌های هسته لینوکس را نمایش می‌دهد.



شکل ۴-۲: پیمانه‌های هسته لینوکس

در دنیای متن‌باز همه قادر به تغییر یا توسعه کارایی هسته لینوکس از طریق به کارگیری پیمانه‌ها هستند.

۳-۲-۲- وصله‌های هسته لینوکس

همان‌گونه که پیمانه‌ها مکانیزمی برای افزودن کارایی‌های جدید به هسته لینوکس فراهم می‌کنند، وصله‌ها نیز راهی برای گسترش یا تغییر هسته ارائه می‌دهند.

وصله‌ها را می‌توان مکانیزمی برای ترمیم و بازسازی کاستیها در نظر گرفت. در مورد لینوکس وصله‌ها برای تغییر دادن یا اضافه کردن کارآیی به هسته نیز به کار می‌روند. مجدداً در قیاس با هوایپیما که کنترلهای راهبری و کنترل جریان بنزین، معادل راهانداز بودند، وصله‌ها در مواردی موتور را تعمیر می‌کنند و در مواردی راههایی برای پیاده‌سازی عملکردهای جدید مثل اضافه کردن قابلیت خنک‌سازی به موتور را فراهم می‌کنند.

عمولاً در حین گسترش لینوکس در سازمان خود متوجه می‌شوید که برنامه‌های کاربردی جدیدی برای کار روی لینوکس نوشته می‌شود. لذا باید در زمینهٔ وابستگیها^۱ چه در مورد نسخهٔ هسته و چه در مورد وصله‌ها و پیمانه‌هایی که احتیاج به نصب دارند، دقت لازم را لحاظ کنید.

۴-۲-۲- تکثر هسته

یکی از بزرگترین نگرانیهای تازه‌واردها به دنیای لینوکس، مسأله تکثر است. تکثر به معنی پیاده‌سازی‌های گوناگون و پراکنده از هسته است.

این مسأله از جنگهای تاریخی یونیکس نشأت می‌گیرد. در اصل یونیکس به منظور فراهم کردن سامانه‌ای با مجموعه‌ای از واسطه‌ای برنامه‌نویسی رایج شکل گرفت. برای انتقال برنامه‌های کاربردی از یک سامانه یونیکس به سامانه‌ای دیگر تنها نیاز به ترجمه مجدد آن روی سامانه مقصد بود. به هر حال عرضه کنندگان یونیکس نیاز به محصولات متنوع داشتند. بنابراین هر عرضه کننده، کارآیی جدیدی به سامانه عامل اضافه می‌کرد و یا کارآیی موجود را توسعه می‌داد که در اکثر موارد این عمل به روش‌های متفاوتی صورت می‌گرفت. نتیجه این شد که با اینکه سامانه‌های یونیکس ریشه و عملکرد مشترکی داشتند، ولی یونیکس‌هایی که عرضه می‌شدند آنقدر متفاوت بودند که کاربر برای انتقال برنامه‌های کاربردی از یک نسخه به نسخه دیگر، به دردرس می‌افتد.

این تفاوتها باعث به قفل شدن^۲ کاربران نیز می‌شد و هزینه‌ها را هنگام تعویض عرضه کننده بالا می‌برد. تولیدکنندگان نرم‌افزار کاری ناشی از تفاوت این محصولات مواجه بودند و نیاز داشتند هزینه‌های انتقال و پشتیبانی هر بستر رایانه‌ای جدید را ضمن دراختیار گرفتن بیشترین سهم بازار، تعدل کنند. هر بستر پشتیبانی شده به پیکربندی و آزمایش گسترهای نیاز دارد. بنابراین تکثر هسته هزینه‌ها را هم برای تولیدکنندگان نرم‌افزار و هم برای مشتریان به طور چشمگیری افزایش می‌دهد.

زمانی که شرکتها و تولیدکنندگان نرم‌افزار کار خود را با لینوکس آغاز کردند و نام تعداد زیاد توزیعهای تجاری لینوکس به گوش آنها رسید، ترس از جنگهای یونیکسی آنها را فرا گرفت. به هر حال همه توزیعهای لینوکس در هسته لینوکس مشترک هستند و این مسأله‌ای است که لینوکس را متفاوت می‌سازد. هسته مشترک لینوکس از نگرانیهای مربوط به هزینه‌های انتقال و

¹ Dependency

² Lock-in

پشتیبانی می‌کاهد و در مقایسه با مسایلی که تولیدکنندگان نرمافزار در گذشته هنگام کار با یونیکس مواجه می‌شدند، برای آنها تسهیلات بیشتری از نظر هزینه و عرضه در بازار به همراه دارد.

اگر چه همه عرضه‌کنندگان با یک هسته مشترک کار می‌کنند، وصله‌ها و خصوصیات دیگر می‌توانند تفاوت‌هایی در میان توزیعهای لینوکس بوجود آورند. فصل هفتم شما را با استانداردهای وضع شده در این زمینه که این اختلافات را جبران می‌کند، آشنا خواهد نمود.

هسته لینوکس متن‌باز است. این حقیقت، از انگیزه عرضه‌کنندگان برای ارائه محصولات مختلف که در سطح پائین زنجیره ارزش قرار دارند، می‌کاهد. در فصل پنجم وقتی توزیعهای لینوکس مفصلًاً شرح داده می‌شوند، نگاه اجمالی به زنجیره ارزش و نحوه تفاوت ارزش محصولات، خواهیم داشت.

زیبایی متن‌باز در این است که اکنون عرضه‌کنندگان انگیزه بالایی برای همکاری و مساعدت با جامعه متن‌باز برای گسترش دادن کارآیی کل سامانه، جهت دستیابی به اهداف بالای خود، دارند. عرضه‌کنندگان نرمافزار در عین اینکه هزینه‌های توسعه را کاهش می‌دهند، در جلب رضایت مشتریان می‌کوشند.

تولیدکنندگان می‌توانند از یک زیربنای مشترک برای توسعه و ایجاد برنامه‌های کاربردی خود استفاده کنند. در نهایت شرکتها بجای ارائه کارهایی دارای عملکردهای سطح پائین و انحصاری، می‌توانند از سامانه مشترک بین عرضه‌کنندگان سود برند و انرژی خود را بر سایر نیازهای تجاری خویش متوجه کنند.

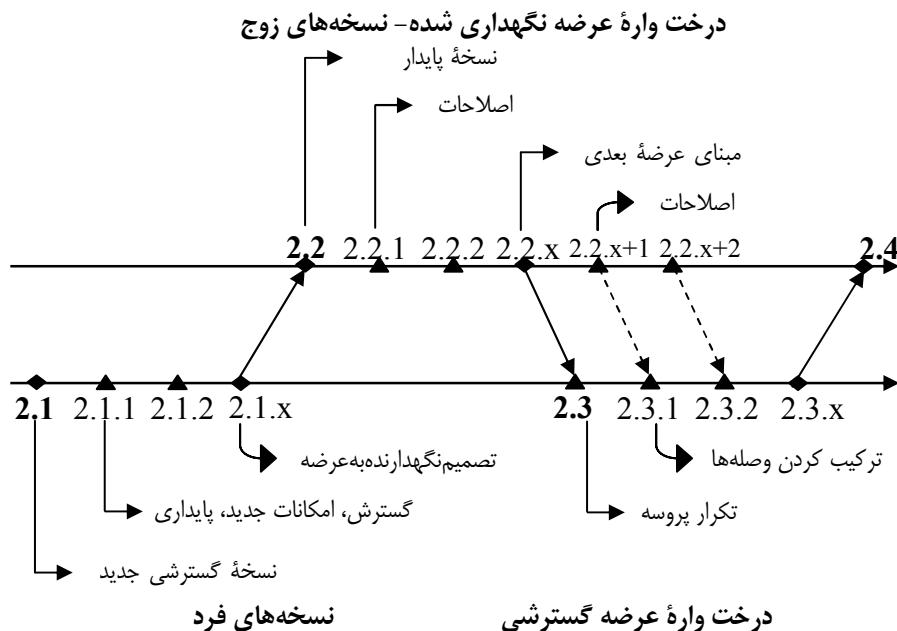
گرچه متن‌باز بودن لینوکس، پراکنده‌گی هسته را غیر محتمل می‌سازد، ولی شرکتها و تولیدکنندگان به مدیریت برخی پیچیدگیها در محیط لینوکس نیاز دارند. همان‌طور که هر سامانه‌ای توسعه می‌یابد، هسته نیز گسترش یافته است و اکنون چندین نسخه از آن موجود است. تولیدکنندگان نرمافزار، توسعه‌دهندگان غیررسمی و دیگر ارائه‌دهندگان نرمافزارها و صله‌هایی برای هسته طراحی می‌کنند و لازم است در زمینه مدیریت وابستگیها چاره‌ای اندیشه شود. بنابراین گونه‌های زیادی از پیمانه‌های هسته (به صورت راهندازها) وجود دارد که ممکن است حین استفاده از لینوکس به آنها احتیاج داشته باشید. نهایتاً عرضه‌کنندگان توزیع ترکیب خاصی از نسخه‌های هسته، بسته‌های نرمافزاری و بهبودهای صورت گرفته روی آنها را ایجاد کنند. بعداً در فصلهای پنجم و ششم و هفت به توزیعهای استانداردها و فرآیندهای توسعه‌ای که امکان مدیریت این پیچیدگی را می‌دهند، خواهیم پرداخت.

۲-۳- طراحی هسته لینوکس و کنترل نسخه‌ها

لینوس توروالدر مالک نشان تجاری ثبت شده لینوکس است. به عبارت دیگر او طراحی نسخه‌های جدید لینوکس را تأیید و تصویب می‌کند. به هر حال مرکز جهانی توسعه هسته لینوکس، Kernel.org است که یک فرآیند توسعه باز و مبتنی بر همکاری را هدایت می‌کند.

به نسخه‌های جدید هسته عدد فردی به عنوان عدد زیر نسخه اختصاص می‌دهند. نسخه‌های هسته 2.1.x، 2.3.x و 2.5.x همه هسته‌هایی هستند که طراحی شده و یا در حال طراحی هستند که می‌توانید آنها را مانند نرم‌افزار ناپایدار کیفیت سنجی آلفا و بتا در نظر بگیرید. قاعدهاً هیچ شرکتی محصول تولیدی خود را بر مبنای این هسته‌ها گسترش نمی‌دهد. نسخه‌های هسته با زیر شماره‌های زوج 2.0.x، 2.2.x و 2.4.x نیز هسته‌های طراحی شده‌ای هستند که آماده استفاده‌اند.

شکل ۵-۲ این سامانه نسخه‌گذاری را نمایش می‌دهد.



شکل ۵-۲: چرخه توسعه هسته لینوکس

بعد از این که هسته طراحی و تصویب شد، لینوس توروالدز چرخه توسعه جدیدی را آغاز می‌کند (زیرشماره فرد بعدی اعلام می‌شود). در این هنگام انجمن هسته برای تعیین لیستی از ویژگیهای که نسخه بعدی هسته گردهم خواهند آمد. یکی از نگرانیهای دنیای لینوکس، نداشتن طرح از پیش تعریف شده‌ای برای آینده و همچنین نداشتن تاریخ قطعی عرضه^۱ است. در جهان توسعه سنتی، طرحهای از پیش تعیین شده، کارآییهای آینده محصول را در هر زمان مشخص می‌کنند. این حقیقت که طرح خاصی برای لینوکس موجود نیست، باعث نگرانی بسیاری از مدیران می‌شود.

^۱ Committed release dates

در این انجمن، تصمیم‌گیرندگان، در عین خبره بودن، به خواسته‌های مشتری نیز اهمیت می‌دهند. بزرگترین تشویق برای یک برنامه‌نویس این است که شاهد به کارگیری گسترده نرمافزار خود باشد. مرحله دیگر عرضه محصول در زمانی است که انجمن آن را آماده تشخیص دهد، نه وقتی که واحدهای بازاریابی یا مدیران غیرمرتب تشخیص دهنند.

اغلب شرکتها، نرمافزار را به واسطه نیاز بازار، فشارهای رقابتی و خواسته‌های مشتری طراحی می‌کنند. بنابراین سازمانهای مهندسی به خاطر الزام به تحويل کار در زمان مقرر، مجبور به استفاده از معماریها و طراحیهای نه چندان مناسب می‌شوند. یکی از ضعفهای این مدل این است که در اکثر موارد، محاسبات و تلاشهای طراحی و معماری، با به پایان رسیدن زمان توسعه به اتمام می‌رسد و این از کیفیت و کارآیی سامانه می‌کاهد. از آن جایی که هدف جامعه متن‌باز، ارائه یک سامانه با کیفیت است و تأکیدی بر بازگشت سرمایه‌های اقتصادی وجود ندارد، متناظراً این چرخه مخرب توسعه هم شکل نمی‌گیرد. این مسئله به این معنی نیست که انجمنها نسبت به نیازهای بازار و درخواستهای مشتریان بی‌توجه هستند، بلکه منظور این است که فرآیندهای طراحی، عقلانی‌تر و با روند منطقی تری شکل می‌گیرند که از اثرات منفی به دور است. این چرخه طراحی نرمافزار، عمدترين دليل اين است که پروژه‌های متن‌باز با سرعت بیشتری نسبت به همتایان تجاری خود رشد می‌کنند و پایداری بیشتری دارند.

بهترین راه برای شتاب دادن طراحی هسته جدید لینوکس و یا اثربخشی بر مجموعه مشخصه لینوکس خیلی ساده است: خود را درگیر کنید! انجمن توسعه لینوکس به صورت ایجاد و حفظ ارتباط و تعامل کار می‌کند. با آنها همکاری کنید و در تیم توسعه لینوکس شرکت کنید. این یک معامله برنده (پایاپای) برای شما و انجمن هسته است. قبول مسئولیتهای کوچک و پشت سرگذاشتن مراحل اولیه، شما را از نزدیک با نحوه پیشرفت فرآیند طراحی و توسعه و زمان احتمالی عرضه نسخه‌های جدید آشنا می‌کند. این همکاری به طراحی بهتر سازمان و زیرساخت فناوری اطلاعات آن منجر خواهد شد. اگر چه لینوکس و دیگر پروژه‌های متن‌باز طرح و قالب منتشر شده‌ای ندارند، اما طبیعت توسعه باز، دید روشنی از فرآیند توسعه در اختیار شما قرار می‌دهد. این دید وسیع باعث می‌شود بتوانید تحلیل بهتر و نزدیک به واقع بینانه‌تری از آینده، نسبت به طراحیهای از پیش چاپ شده، بدست آورید.

از آنجایی که هسته‌های توسعه داده شده به ندرت در محیط تولید آزمون می‌شوند، مشخص کردن همه کاستیهای سامانه ممکن نیست. گرچه جامعه متن‌باز بهترین آشکارکننده کاستیهای است، در عمل استفاده از یک محیط تولید مشارکتی بهترین راه برای رفع کاستیهای یک سامانه است. این فرآیند ممکن است خطاهایی را آشکار کند که نیاز به تصحیح دارند. وقتی لینوس توروالدز یک نسخه نامگذاری شده با شماره‌های زوج هسته لینوکس را طراحی می‌کند، مسئولیت نگهداری آن به "مارچلو توساتی"^۱ سپرده می‌شود (این مسئولیت را چندین سال آن کوکس

¹ Tosatti

بر عهده داشت). همان طور که شرکتها لینوکس را استفاده می‌کنند و کاستیها را آشکار می‌سازند، "مارچلو" تغییرات لازم را روی هسته اعمال می‌کند. لینوس و مارچلو با هم در ارتباط مستمر هستند و این خیلی مهم است که مارچلو تغییراتی را که از لحاظ معماری اثرات منفی بلندمدت روی هسته می‌گذارند، پیاده‌سازی نکند. قرارداد و اطمینان بین لینوس و مارچلو و آمادگی آنها برای پذیرفتن پیشنهادهای از طرف انجمن توسعه متن‌باز، به گونه‌ای است که تحقق این اتفاق بعید به نظر می‌رسد.

وقتی شما با شرایط ناخواسته در گسترش لینوکس مواجه می‌شوید، عملی‌ترین راه برای مقابله با آنها، ایجاد یک رابطه قوی با انجمن توسعه‌دهندگان هسته است. اغلب افرادی که با این کاستیها روپرتو می‌شوند، نه تنها مشکل را مستند می‌کنند، بلکه آن را به صورت دیگری مثل نیز درخواست اصلاح کد، به مارچلو توسعه ارائه می‌دهند.

طمئن‌نمای شرکتهای زیادی وجود دارند که کارمندان فناوری اطلاعات^۱ آنها از تعداد، تجربه و دانش لازم برای کارشناسی هسته برخوردار نیستند. در این موقع شرکتها عموماً از عرضه‌کنندگان لینوکس برای پشتیبانی کمک می‌گیرند. ما این مدل‌های پشتیبانی را با تفصیل بیشتر در فصل هشتم بررسی خواهیم کرد.

وقتی طراحی هسته به حد قابل قبولی می‌رسد، این هسته پایدار به عنوان پایه‌ای برای توسعه جدید هسته به کار خواهد رفت و این چرخه تکرار خواهد شد. وقتی که یک هسته تولیدی و یک هسته توسعه‌ای هر دو در حال گسترش هستند، مارچلو و لینوس برای اطمینان از اینکه هم گامی آنها حفظ می‌شود، باهم کار می‌کنند.

۲-۳-۱-۲- پشتیبانی چند بستر^۲

یکی از نقاط قوت لینوکس از این واقعیت نشأت می‌گیرد که لینوکس گستردگرترین سامانه‌عامل قابل انتقال به بسترهای دیگر^۳ است. لینوس توروالدز در ابتدا هسته را به عنوان پروژه‌ای برای درسی دانشگاهی طراحی کرد. چرا که او سامانه‌عامل شبه یونیکسی، برای رایانه شخصی مبتنی بر اینتل ۳۸۶ جدیدش می‌خواست و قابلیت انتقال هدف وی نبود. در ابتدا هسته به طور مستقیم با ریزپردازنده ۳۸۶ مرتبط بود و همانند آنچه در مورد اکثر پروژه‌های متن‌باز رخ می‌دهد، کمتر کسی می‌توانست روند رشد این پروژه را پیش‌بینی کند.

توسعه‌دهندگان زیادی به قابل انتقال کردن هسته لینوکس روی آورده‌اند و سعی بر ارائه ساختار مناسب داشته‌اند. امروزه درخت توسعه هسته لینوکس هم انشعبابات مستقل از معماری و هم انشعبابات وابسته به معماری دارد. انشعباب مستقل از معماری و کد مربوط به معماری اینتل x86 همگی بوسیله لینوس نگهداری می‌شوند (مؤلفه‌هایی مثل درایورها و ریزپردازنده‌های دیگر توسعه

¹ Information Technology

² Platform

³ Portable

گروههای جداگانه‌ای از توسعه‌دهندگان نگهداری می‌شوند). وقتی این توسعه‌دهندگان با مشکلات مربوط به پردازندۀ مواجه می‌شوند، می‌توانند تغییرات لازم را برای اطمینان از کار کردن صحیح سامانه اعمال کنند. در برخی موارد لازم است که تغییرات به مؤلفه‌های مستقل از معماری اعمال شود که به تبع آن باید تغییراتی به قسمت تحت کنترل لینوس نیز وارد شود. نگهدارندگان هسته برای طراحی و اعمال این تغییرات در رابطهٔ تنگاتنگی با لینوس کار می‌کنند.

۲-۳-۲ - بازار بزرگ در مقابل پردازندۀ‌های نامشهور

شرکتهایی که توسعه لینوکس را از طریق ریزپردازندۀ‌های نامشهور دنبال می‌کنند، باید فاکتورهای اضافه‌ای در تصمیم‌گیریهای خود لحاظ کنند. به یاد آورید که یکی از مزایای اصلی لینوکس این است که لینوکس اقتصاد کالایی (عمومی شده) را به سامانه‌های عامل تعمیم می‌دهد. الحقیقت ریزپردازندۀ‌های کالایی (عمومی شده) به بازار لینوکس آن را به گزینه‌ای مقرون به صرفه و بالرزش برای تجارت بدل می‌سازد.

عدد زیادی قابل انتقال بودن لینوکس را یکی از توانمندیهای لینوکس می‌دانند. در مواردی ممکن است به کارگیری لینوکس در پردازندۀ‌های اختصاصی گزینه مطلوب و ارزشمندی باشند. مثلاً وقتی خصوصیات، کارآیی یا مجموعه ویژگیهای خاصی مد نظر باشند که پردازندۀ خاصی از عهده آنها بهتر برآید. به کارگیری لینوکس تحت ریزپردازندۀ‌های نامشهور نیز ایده ارزشمندی است به خصوص وقتی که شما نیاز به انتقال از یک بستر یا سامانه‌عاملی دارید و می‌خواهید از پردازندۀ‌های اختصاصی به پردازندۀ‌های مشهور روی آورید. بعضی از عرضه‌کنندگان توزیعهای لینوکس، محصولات لینوکس خود را روی پردازندۀ‌های نامشهور تولید نموده که این فرآیند را تسهیل می‌کند. هنگام ارزیابی پردازندۀ‌های جایگزین موارد زیر را لحاظ کنید:

- هم اکنون توسعه اصلی لینوکس روی معماری $x86$ انجام می‌شود.
- برنامه‌های کاربردی مورد نیاز برای کار شما ممکن است موجود نباشند و در مدت کوتاهی حاضر شوند.
- تولیدکنندگان نرمافزار ممکن است از پشتیبانی برنامه‌های کاربردی خود روی هر سامانه یا پردازندۀ غیر متداول سرباز زنند. آنها ممکن است به منابع لازم برای پشتیبانی از معماری‌های دیگر دسترسی نداشته باشند.
- جامعه توسعه‌دهندگان متن‌باز به ندرت به منابع مورد نیاز برای توسعه و آزمون برنامه‌های خود روی معماری‌های دیگر دسترسی دارند.
- تغییراتی که لازم است در اجزای وابسته به معماری هسته اعمال شود، ممکن است آن را برای افراد دیگر نامطلوب سازد.
- اکثر نویسندهای راهاندازها امکان آزمون کردن دستگاه‌های خود را روی معماری‌های دیگر ندارند.
- آموزش‌دیدگان لینوکس، به ندرت با لینوکس روی معماری‌های دیگر آشنایی دارند.

گرچه این لیست فهرست مهمی از نکاتی است که باید مدنظر قرار داد، با این حال ممکن است شما قصد گسترش روی پردازنده‌های نامشهور را داشته باشید. امروزه هسته لینوکس علاوه بر معماری پردازنده x86، بر روی معماری‌های زیر نیز موجود می‌باشد:

- AMD از Opteron •
- HP از PA-RISC •
- IBM از PowerPC •
- IBM از S/390 •
- خانواده از Itanium •
- Motorola از 68K •
- SUN از SPARC •

میزان پشتیبانی در هر مورد متفاوت است. در اکثر موارد توزیعهای کامل لینوکس (فصل ۵) موجود هستند.

۲-۴- لینوکس روی میز کار

در حال حاضر گسترش لینوکس روی میز کار، مورد بحث محافل زیادی است. آمارها نشان می‌دهد لینوکس روی میز کار نفوذی کمتر از ۲ درصد و به گفته بعضی آمارها کمتر از نیم درصد داشته است. [این کار مربوط به سال ۲۰۰۳ در زمان تأثیف کتاب است. در حال حاضر پروژه‌های بزرگ مهاجرت در بسیاری از نقاط دنیا انجام شده (اکسترمادورا در اسپانیا) یا در حال انجام است. (شهرهای وین، مونیخ، برلین)] طرفداران لینوکس ادعا می‌کنند که لینوکس برای مطرح شدن به عنوان سامانه‌عامل رومیزی ارزشمند، نیاز به زمان دارد.

مهمنترین اثبات این ادعا، شهرت لینوکس در اغلب کشورهای جهان است. در دسترس بودن برنامه‌های کاربردی، قابلیت جایگاهی از یک بستر یا سامانه‌عامل به دیگری و هزینه‌های آموزش مجدد، گسترش لینوکس روی میز کار بسیاری از سازمانها را با مشکل مواجه کرده است.

بخشهای کلیدی از بازار وجود دارند که لینوکس می‌تواند گرینه مناسبی برای میز کار باشد. شرکتهای برنامه‌نویسی که قبلاً از ایستگاه‌های کاری یونیکس استفاده می‌کردند، جایگاه مناسبی برای میز کارهای لینوکس هستند. همچنین برنامه‌های کاربردی پردازش تصویر و پویانگاری مثل صنعت بازیها، سرگرمیها و تصویربرداری پیشکی نیز کاندیداهای مناسبی برای گسترش لینوکس هستند.

توسعه‌دهندگان نرم‌افزار نیز جذب لینوکس رومیزی شده‌اند. زیرا آنها می‌توانند با استفاده از رایانه‌های شخصی کم هزینه مبتنی بر لینوکس، برنامه‌های کاربردی را برای محیط‌های یونیکس

گسترش دهنده در نهایت ایستگاههای کاری مالی که برای کاربردهای خاص مثل خودپردازها استفاده می‌شوند، می‌توانند رویکردهای مناسبی برای میزکار لینوکس باشند. همه موارد فوق پیشینه مشترکی در استفاده از ایستگاههای کاری یونیکس دارند. در حقیقت لینوکس می‌تواند با کمک یک رایانه شخصی ارزان قیمت، تجربه یونیکس را در اختیار کاربران پیشرفت‌هه قرار دهد. انجمن متن‌باز دو واسط کاربری رومیزی[گرافیکی] مشهور ارایه داده است: محیط رومیزی GNOME و محیط رومیزی K (یا KDE). برای سادگی می‌توانید آنها را مانند داشبورد ماشین خود فرض کنید.

اگر با میزکارهای یونیکس آشنا باشید می‌توانید KDE و GNOME را معادل CDE در یونیکس در نظر بگیرید. هر محیط رومیزی، «حس و ظاهر» یکتای خودش را دارد و واسط برنامه‌نویسی خاصی را پشتیبانی می‌کند. این واسط برای Motif، CDE، GNOME، برای Qt، KDE و برای Star Office می‌باشد. یکی از زیباییهای لینوکس این است که صرف‌نظر از محیط رومیزی که انتخاب می‌کنید، می‌توانید برنامه‌های کاربردی خود را روی هریک از این واسطه‌های برنامه‌نویسی اجرا کنید.

شرکت شما می‌تواند یک محیط رومیزی خاص را انتخاب کند و یا اینکه به کاربران خود اجازه انتخاب دهد و آنها را از اجرا شدن برنامه‌های کاربردی خود مطمئن سازد.

تعدادی برنامه کاربردی سودمند اداری برای میزکارهای لینوکس موجود است. مشهورترین آنها نرمافزاری Star Office می‌باشد که توسط SUN ارائه شده است و Open Office که نسخه متن‌باز Star Office است و توسط SUN و جمعی از توسعه‌دهنگان متن‌باز نگهداری می‌شود.

۲-۵- مقیاس پذیری عمودی و افقی

هسته 2.4 گام بلندی برای مقیاس‌پذیری چندپردازنهای (یا عمودی) بود. بسته به کاربرد، لینوکس می‌تواند به خوبی تغییر اندازه دهد و گروههای صنعتی و کاری با این روند به جلو می‌روند. هسته لینوکس نوعاً برای کار تحت یک پیکربندی چندپردازنه و یا تک پردازنه ترجمه می‌شود. پیکربندی چندپردازنه مقیاس‌پذیری عمودی را ممکن می‌سازد. قبل از شکل‌گیری هسته 2.4 تنها روش مقیاس‌پذیری، گروهی (یا افقی) بود. یکی از فناوریهای گروهی در حال توسعه از سال ۱۹۹۴ Beowulf می‌باشد. شکل ۲-۶ تفاوت بین مقیاس‌پذیری عمودی و افقی را نشان می‌دهد.



شکل ۶-۲: مقایس پذیری افقی و عمودی

برنامه‌های کاربردی زیادی که مبتنی بر پردازش داده سنتی بودند، اکنون هسته و سامانه‌عاملی دارند که مقایس دهی عمودی را برای دربرگرفتن طیف وسیعی از نیازمندیها فراهم می‌سازد. مواردی که به پردازش موازی حجیم نیاز داشتند نیز بیشتر از ۸ سال است که صاحب این فناوری شده‌اند.

پروژه دانشگاهی لینوس توروالدز، اکنون در مدت زمان کوتاهی به سطح [بالایی] از کاربرد و پیچیدگی - معادل با آنچه توسط سامانه‌های تجاری یونیکس عرضه شده بود - رسیده است.

۲-۶- لینوکس توکار^۱

شرکتهای بسیاری وجود دارند که در ارایه یک لینوکس برای کاربردهای توکار (مانند کنترل اتوموبیل) تخصص دارند. هسته لینوکس قابلیت خوبی برای برنامه‌های کاربردی توکار دارد: مثلاً می‌توان یک هسته لینوکس با کارآیی محدود و مقایس کوچک ساخت که با یک حافظه یک مگابایتی کار کند.

با توجه به مجوز لینوکس، می‌توان آن را به طور رایگان کپی برداری کرد. از آنجایی که هر دستگاه تولیدی نیازمند یک برنامه کاربردی توکار است، استفاده از لینوکس باعث می‌شود میلیونها دستگاه تولید شود، بدون اینکه نیازی به پرداخت حق الامتیاز باشد.

همچنین شما می‌توانید از کار شرکتهای دیگر به ویژه آنها بیکار که در زمینه هسته‌های توکار لینوکس کار می‌کنند، استفاده نمائید. اگر شما نیازمندیهای توکار پیچیده یا خاص دارید، می‌توانید

¹ Embedded

یک هسته توکار دلخواه خود طراحی نموده یا از خدمات سایر شرکتهای استفاده کنید. تصمیم‌گیری در این زمینه به تواناییهای شما در پشتیبانی از محیط مورد نظر بستگی دارد.

۷-۲- جمع بندی

شما در این فصل یکی از پیچیده‌ترین موضوعات در علم رایانه را بررسی کردید. درک هسته لینوکس یکی از پایه‌های کلیدی برای فهم نحوه شکل گرفتن یک سامانه لینوکس کامل و چگونگی گسترش آن در سازمان است.

وقتی در فصل پنج توزیعهای لینوکس بررسی می‌شوند، خواهید دید که هسته چگونه به سامانه بزرگتر متصل می‌شود.

در فصل بعد، سعی خواهیم کرد درک شما را از قانون مالکیت معنوی گسترش دهیم و نمونه جدیدی در فرآیندهای توسعه نرمافزار را بررسی کنیم. متن باز بودن یکی از دلایلی است که توسعه لینوکس و برنامه‌های کاربردی دیگر را ممکن می‌سازد و فهم نحوه کار پروژه‌های متن باز و زیربنای قانونی آن، پیش نیاز توسعه لینوکس است.

فصل سوم:

متن باز - هدایت مسیر قانونی به آزاد بودن

پیشرفت سریع لینوکس سبب شده است که بتواند جایگاه ویژه‌ای در صنعت پیدا کند. لینوکس تنها پروژه متن باز عظیمی است که صدها توسعه‌دهنده در ارتباط با آن کار می‌کنند و قابلیتهای متن باز را به خوبی به نمایش گذاشته است. بنابراین شما به عنوان مدیر فناوری باید در زمینه متن باز خبره باشید. این متادلوزی جدید توسعه نرم‌افزار و مدل مجوزدهی آن، جدیدترین فرصتها و بزرگترین تهدیدها را برای سازمان شما به همراه دارد. لینوکس فرصتهای جدیدی فراهم می‌کند، زیرا راههای جدیدی برای صرف هزینه‌های کمتر و ایجاد مزایای رقابتی پیش رو می‌نهد و تهدیدهایی به دنبال دارد. چرا که حرفه و تجارت را درگیر یک سری خطر می‌کند. از آنجایی که توسعه‌دهنگان زیادی در سازمان از منابع نرم‌افزاری متن باز استفاده می‌کنند، شما باید در این باره توجه و دقت کافی داشته باشید. مشکل دیگر اینجاست که همین توسعه‌دهنگان، به طور کامل به آفات متن باز اشراف ندارند. شما می‌توانید با خبره شدن در این زمینه، در حالی که از مزایای فرصتهای پیش رو بهره می‌برید، ریسکها را حذف کنید. اهداف این فصل عبارتند از بررسی موارد زیر:

- آزادی بودن و متن باز
- تعریف متن باز^۱
- مجوزهای متن باز شامل GPL
- مجوزهای غیر متن باز که اغلب با مجوزهای متن باز اشتباہ گرفته می‌شوند
- متادلوزی توسعه متن باز

این فصل بر تشریح متن باز و حرکت قانونی آن تمرکز دارد. بخش سوم این کتاب بحثهای کاملی روی فرآیندهای تجاری مرتبط با استفاده و متن باز شرکت نموده و راهنماییهای ارائه می‌دهد که به شما کمک می‌کند بتوانید درباره زمان پیاده‌سازی نرم‌افزار متن بازی که توسعه داده‌اید، تصمیم [صحیحی] [بگیرید].

۳-۱- آزادی متن باز بودن

ریشه متن باز از نرم‌افزار آزاد^۲ گرفته شده است. اکثر مردم وقتی برای اولین بار واژه نرم‌افزار آزاد را می‌شنوند، فکر می‌کنند این واژه به معنای نرم‌افزار رایگان است. این تصور درستی نیست. واژه Free وقتی در مورد نرم‌افزار به کار گرفته می‌شود به مفهوم آزاد بودن است – آزاد بودن در نسخه‌برداری، تغییر دادن و توزیع نرم‌افزار. در اصل آزاد بودن در این زمینه‌ها مستلزم داشتن دسترسی یکسان به کد منبع است. واژه Free Software ساخته ریچارد استالمن است. او نرم‌افزارهای مهمی برای یونیکس طراحی کرد و آنها را تحت مجوز GPL خود در اختیار عموم

¹ Open Source Definition (OSD)

² Free Software

قرار داد. سپس "بنیاد نرم افزار آزاد" یا FSF را برای ترویج این برداشت از آزادی، ایجاد کرد. طبق مدلی که توسط FSF ایجاد شد، لینوکس را تحت مجوز نرم افزار آزاد GPL توزیع کرد.

دیگر پیشگامان حرکت نرم افزار آزاد مثل "اریک ریموند" و "لاری آگوستین" نسبت به برداشت اشتباه از معنی کلمه Free در انگلیسی نگران بودند. چرا که آنها معتقد بودند معانی دیگر این کلمه (مثل رایگان) شرکتهای بزرگ نرم افزاری را بدون دلیل نسبت به این مدل توسعه نرم افزار به وحشت می اندازد. ریموند اقدام به نوشتن مقاله تاریخی خود با عنوان "کانترال و بازار" در شرح و توصیف طراحی نرم افزارهای با کیفیت توسط جوامعی از برنامه نویسان، نوشته است. او برای توصیف این نرم افزارها واژه "متن باز" را ابداع کرد، چرا که منطبق بر هیچ مدل اقتصادی نبود. هنوز هم دسترسی به کد منبع شرط لازم برای این فرآیند طراحی و توسعه محسوب می شود.

همچنین این دسته از پیشگامان به این نتیجه رسیدند که موفقیت متن باز وابسته به توصیفی روش از آنچه که در مجوز متن باز آمده است، می باشد. بروس پرنز، مدیر پروژه انجمن Debian^۱ برای عقد یک قرارداد اجتماعی با انجمن نرم افزار آزاد تلاش فراوانی کرد. (در مورد دیان در فصل ۵ بطور مفصل صحبت خواهد شد فعلاً به همین اکتفا می کنیم که دیان یکی از توزیعهای لینوکس است که توسط جامعه برنامه نویسان متن باز گسترش یافته است). او و دیگر پیشگامان حرکت متن باز توافق کردند که این قرارداد به عنوان "تعریف رسمی متن باز" یا OSD برگزیده شود. به همین دلیل است بروس پرنز را به عنوان طراح OSD می شناسند.

قابل ذکر است که افرادی مثل ریچارد استالمان با تغییر نام به متن باز موافق نبودند. علت این امر را در قسمتهای بعدی این فصل، وقتی GPL و مجوزهای دیگر را تشریح می کنیم، بررسی خواهیم کرد.

۳-۲- تعریف متن باز

به زبان ساده مجوز متن باز حقوق هر کس را در هر جا با هر هدفی شامل استفاده، نسخه برداری، تغییر و توزیع نرم افزار محافظت می کند. در عمل این کار مستلزم دسترسی آزادانه به کد منبع است.

این حقوق الزامی به طور رسمی در OSD تدوین شده اند و توسط OSI^۲ منتشر شده اند (OSI یک موسسه غیرانتفاعی است). انجمن متن باز، OSD را به عنوان مرجعی برای تشخیص مجوزهای واقعی متن باز می شناسد. مجوزهایی که با مطابق OSD ارائه می شوند، ملزوماتی را که در بالا شرح داده شد، رعایت می کنند.

یک مسئله خیلی مهم که در اینجا مطرح می شود این است که OSD خودش یک مجوز نیست. نویسنده اولین نسخه OSD از آن این گونه یاد می کند: ویژگیهایی که یک مجوز واقعی

¹ Debian

² Open Source Initiative

باید داشته باشد تا نرم افزارهایی که تحت این مجوز توزیع می شوند، بتوانند از نظر قانونی متن باز باشند. جزئیات بیشتری از OSD را می توانید در قسمت مرجع در انتهای کتاب بیابید.
یک نشان اعتباری به عنوان شاخص "متن باز قانونی" طراحی کرده است. وقتی روی یک OSI بسته نرم افزاری "OSI Certified" را می بینید، می توانید مطمئن شوید که مجوزی که آن نرم افزار تحت آن توزیع شده است، با اصول OSD سازگار است.

چنین تعریف استانداردی برای مجوزهای اختصاصی وجود ندارد. هر شرکتی مجوز مربوط به خود را طراحی و استفاده می کند. در موقع کار با هر یک از این گونه مجوزها، لازم است یک حقوقدان شرایط آن را با دقت مطالعه کند و همواره سازمان را در زمینه رعایت آنها راهنمایی کند.

در سالهای قبل مردم راجع به مفهوم و معنای متن باز سوالات متعددی از من می پرسیدند. من به این نتیجه رسیدم که بهترین جوابی که می توان به آنها داد خلاصه ای از تعریف مطرح شده در OSD است . در زیر خلاصه ای از آن آمده است. اگر تعریف جامع تری می خواهید، به وب گاه^۱ OSI مراجعه کنید. همه این شرایط باید در یک مجوز لحاظ شده باشند تا بتوان آن را متن باز تلقی کرد و نرم افزاری که تحت آن مجوزدهی می شود، تأیید شده OSI باشد. شرایط OSI عبارتند از:

۱. توزیع مجدد رایگان - مجوز متن باز نمی تواند صاحب جواز را از توزیع مجدد نرم افزار به طور رایگان منع کند. صاحب جواز مجاز است که تعیین کند آیا نسخه برداری هزینه دربردارد یا خیر و مقدار این هزینه چقدر خواهد بود. عموماً فراهم کنندگان نرم افزار بابت رسانه، کتاب راهنمای و خدمات پشتیبانی محصول نرم افزاری متن باز هزینه دریافت می کنند. در عمل این شرط منجر به عرضه نرم افزار به صورت رایگان یا با هزینه کم می شود.
۲. کد منبع - کد منبع برنامه باید موجود و در دسترس باشد. در اکثر موارد کد منبع همراه با قسمت باینری (برنامه اجرایی) ارائه می شود، ولی همیشه این گونه نیست. به هر حال اگر کد منبع همراه با دودویی ارائه نشود، باید راهی برای دریافت آن به صورت رایگان موجود باشد. در این حالت چون کد منبع موجود است، هر فردی یا به عبارت صحیح تر هر فردی دارای مهارت های لازم می تواند خطاه را کشف کند یا قابلیت های نرم افزار را افزایش دهد.
۳. کارهای مشتق شده - مجوز باید به نرم افزار اجازه تغییر و به نرم افزار تغییر یافته [مشتق شده] اجازه پخش مجدد بدهد. اگر نرم افزار متن بازی را در اختیار بگیرید، در تغییر دادن آن آزادید و همچنین می توانید نسخه تغییر یافته را مجدداً توزیع کنید.
۴. سالم بودن کد منبع نویسنده برنامه - در روند توسعه در جامعه متن باز، درختواره رسمی کدهای منبع (مجموعه کدهای منبع تولید شده و اصلاح شده مورد تایید) یک پروژه، توسط نگهدارنده آن کنترل می شود. در مواردی ممکن است نگهدارنده با اعمال برخی تغییرات به

¹ <http://www.opensource.org>

درختواره رسمی کدهای منبع موافق نباشد و آنها را رد کند، ولی برنامهنویس خواهان توزیع آن تغییرات است. از نقطه نظر تشخیص و مجوزدهی لازم است که راه مشخصی برای جدا کردن کدهای منبع رسمی از گونه‌های تغییر یافته وجود داشته باشد. شرط جامعیت^۱ در OSD تضمین می‌کند که شما می‌دانید چه کسی مسئول کد منبعی است که از آن استفاده می‌کنید. بنابراین نویسنده اصلی فقط در زمینه کدی که خودش نوشته است تحسین یا سرزنش می‌شود. برخی از مجوزهای متن‌باز ملزم می‌کنند که تغییرات از طریق وصله‌ها اعمال شوند. در این حالت کد اصلی بدون تغییر، توزیع می‌شود و تغییرات در پروندهای جداگانه‌ای ارائه می‌شوند. این دو در زمان ترجمه ترکیب می‌شوند تا نرم‌افزاری با قابلیت‌های جدید ایجاد شود.

۵. نبود تبعیض در برابر اشخاص یا گروه‌ها – مجوز متن‌باز نمی‌تواند استفاده یا توزیع مجدد نرم‌افزار و کد منبع آن را برای هیچ شخص یا گروهی منع کند. این شرط هیچ محدودیتی ندارد، حتی اگر از برخی کاربران نرم‌افزار متنفر هم باشیم (مثلاً نمی‌توانید تروریستها را از استفاده از نرم‌افزار منع کنید). از نقطه نظر تجاری، این شرط این معنی را در بر دارد که نمی‌توانید یک رقیب را از استفاده و توزیع مجدد نرم‌افزار خود منع کنید.

۶. نبود تبعیض در کاربرد – این شرط مانع کنترل یا محدودکردن نحوه استفاده از نرم‌افزار است. صادر کننده مجوز نرم‌افزار نمی‌تواند عقاید فرهنگی، اجتماعی و یا سیاسی خود را روی مجوزهای نرم‌افزار تحمیل کند. مجوز نمی‌تواند مانع تجارت کردن نرم‌افزار شود.

۷. توزیع مجوز – در مدل متن‌باز، تنها مجوزی که می‌توان بر نرم‌افزار تحمیل کرد، مجوز متن‌باز است. لزومی به امضاء موافقتname عدم افشاء اطلاعات^۲ یا صدور مجوز ثانویه برای بدست آوردن نرم‌افزار نیست.

۸. مجوز نباید خاص یک محصول باشد – نرم‌افزارهای متن‌باز نمی‌تواند محدود به استفاده به همراه محصولات خاص شود. مثلاً نرم‌افزارهای متن‌باز محدود به استفاده با لینوکس نیستند.

۹. مجوز نباید با دیگر نرم‌افزارها درگیر شود – برای اینکه مجوزی با OSD سازگار باشد، نباید شرایط خود را بر نرم‌افزارهای دیگر توزیع شده به همراه آن تحمیل کند. این امکان، شرکتها را به توزیع نرم‌افزارهای متن‌باز و اختصاصی روی یک رسانه (مثلاً لوح فشرده)، تشویق می‌کند. این^۳ شرط برای همه مجوزهای متن‌باز ضروری هستند.

۳-۳- مالکیت معنوی و دوطرفه بودن

بعضی از مجوزهای متن‌باز بار مسئولیت دوطرفه را بر صاحب جواز تحمیل می‌کنند. خیلی مهم است که این مفهوم و تأثیرات آن روی مالکیت معنوی به درستی درک شود. ولی اول باید بفهمیم که معنی دوطرفه بودن^۳ چیست و چرا تعداد زیادی از طرفداران نرم‌افزار آزاد اصرار دارند

¹ Integrity

² Non Disclosure Agreement

³ Reciprocity

در مجوزهایشان از این مفهوم بهره گیرند. دوطرفه بودن شما را ملزم می‌کند که تمامی کارهای مشتق شده از کار اصلی را تحت مجوز متن باز مشابهی در اختیار نویسنده اصلی کار و سایر کاربران قرار دهید. مجوزی که لینوکس تحت آن ارائه شده است، GPL نام دارد که شرط دوطرفه بودن را در بردارد.

FSF از واژه کپی‌لفت^۱ برای توصیف دوطرفه بودن در GPL استفاده کرده است. علت این انتخاب نشان دادن رابطه آن واژه با قانون حق نشر^۲ صورت گرفته است. کپی‌لفت روشه عمومی برای تبدیل یک برنامه به نرمافزار آزاد و همچنین ملزم کردن برنامه‌نویسان به توزیع نسخه‌های تغییر یافته آن می‌باشد. برای این کار ابتدا نرمافزار کپی‌لفت می‌شود، سپس تحت یک مجوز عرضه می‌شود، تغییراتی در آن ایجاد شده، کد برنامه تغییر یافته، (برنامه‌های دیگری از آن مشتق می‌شود) [و در نهایت با همان مجوز] توزیع می‌گردد. بنابراین کد منبع و آزادیهای مربوط به آن قانوناً [از نرمافزار و نسخه‌های تغییر یافته آن] جدنشدنی می‌شوند.

دوطرفه بودن کپی‌لفت به طراح یک نرمافزار، امکان بهره‌گیری از تلاش‌های شما را می‌دهد، همان‌طور که شما از مزایای در اختیار داشتن کد منبع [تولید شده دیگران] بهره می‌برید. ارائه‌دهندگان نرمافزار آزاد، انگیزه زیادی برای ارائه محصولات خود تحت این مدل دارند، زیرا می‌دانند که نرمافزار آزاد بهتری بدست می‌آورند.

شرط کپی‌لفت شامل استفاده از نرمافزار نشده و فقط زمانی معنی پیدا می‌کند که نرمافزار تغییر یافته مجدداً توزیع می‌شود. بنابراین اگر شما روی رایانه‌های فروخته شده به مشتریان خود از لینوکس به عنوان سامانه‌عامل استفاده کنید، الزامات دوطرفه بودن شامل حال شما نخواهد شد و این الزامات زمانی اعمال می‌شوند که در زمان استفاده از یک کار مشتق شده، برنامه اصلی را - به همان تفسیری که در قانون حق کپی آمده است - تغییر داده باشید. بنابراین اگر صرفاً لینوکس را همان‌طور که دریافت کردید بین مشتریان توزیع کنید، الزامات کپی‌لفت در مجوز لینوکس شامل حال شما نمی‌شوند.

اگر لینوکس را تغییر دهید و نسخه تغییر یافته را توزیع کنید، باید آن تغییرات را تحت قوانین GPL مجوزدهی کنید. این تنها حالتی است که شما ملزم به رعایت الزامات دوطرفه بودن هستید. وقتی نرمافزاری را که از برنامه‌نویسی دریافت کرده‌اید، تغییر می‌دهید، بهتر است در مورد اینکه آیا این عمل مشمول قاعده کارهای مشتق شده می‌شود یا با یک حقوقدان متخصص امور مالکیت معنوی مشورت کنید. طبق تمامی مجوزهای متن باز دوطرفه، شما باید کار جدید خود را که با اعمال تغییرات در نرمافزار دریافتی ایجاد شده است، تحت همان مجوز متن باز (یا مجوزی مشابه) در اختیار سایرین بگذارید.

¹ Copyleft

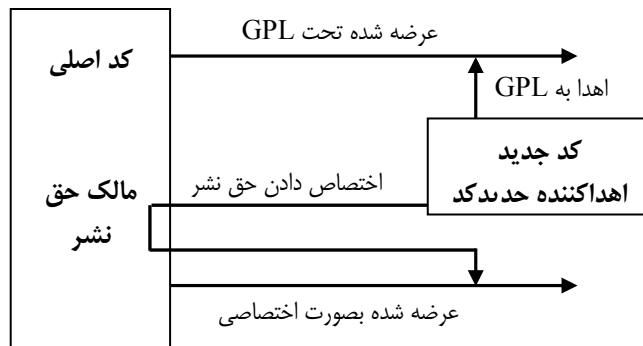
² Copyright

۴-۳- مالکیت حق نشر و مجوز دوگانه

هر دارنده حق نشر می‌تواند نرم‌افزار خود را تحت یک یا چندین مجوز، مجوزدهی کند. بعد از آن هر صاحب جواز (خریدار) می‌تواند انتخاب کند که نرم‌افزار را تحت چه مجوزی بپذیرد. در بعضی از نرم‌افزارها صاحب جواز به شما امکان انتخاب را بین مجوزدهی نرم‌افزار تحت یک مجوز متن‌باز دوطرفه مثل GPL و یا مجوزهای تجاری که باید برای آنها حق الامتیاز پرداخت شود، می‌دهد. چنین استراتژیهای بازاریابی مجوز دوگانه، در پاسخ به آن دسته از مشتریانی توسعه یافته است که تمایلی به پذیرش الزامات دوطرفه بودن نداشته و از آن اجتناب می‌ورزند.

لینوکس تحت یک مجوز دوگانه ارائه نشده است، ولی تعدادی از نرم‌افزارهای متن‌باز مهم که تحت لینوکس اجرا می‌شوند، این گونه‌اند مثلاً "هاینس رایزر"، سامانه‌پرونده خود به نام ReiserFS را تحت چند مجوز ارائه داده است. این امر امکان به کارگیری ReiserFS سامانه‌عامل لینوکس تحت قوانین GPL را می‌دهد. رایزر همچنان می‌تواند سامانه‌پرونده خود را تحت یک مجوز اختصاصی نیز مجوزدهی کند. توسعه‌دهندگان سامانه‌های عامل دیگر با رایزر برای گسترش سامانه‌پرونده وی کار می‌کنند. تفاوت کار در این است که رایزر می‌تواند در مقابل سامانه‌پرونده خود هزینه درخواست کند و توسعه‌دهندگان می‌توانند ساختار پشتیبانی مؤثرتری برای این سامانه‌پرونده بوجود آورند. این مورد یک مثال از موارد بسیاری استفاده از مجوز دوگانه است.

شکل ۱-۳ روند پیاده‌سازی یک مدل مجوز دوگانه را نشان می‌دهد. [نمونه مشهورتر این گونه مجوزدهی بانک اطلاعاتی MySQL است.]



شکل ۳- اختصاص دادن حق نشر در مجوزدهی دوگانه

توسعه‌دهندگان متن‌باز برای اینکه هنکام مجوزدهی دستیشان باز باشد، اغلب از ارایه‌دهندگان [و مشارکت‌کنندگان در توسعه نرم‌افزار] کد درخواست می‌کنند که حق کپی کد را در اختیار آنها قرار دهنند. با این روش توسعه-دهنده اصلی حق دارد در مورد استراتژی مجوزدهی برای نسخه توسعه یافته تصمیم بگیرد.

لینوس توروالدز نیازی به دراختیار گرفتن حق پیشرفت‌های لینوکس را ندارد. ولی FSF آماده پذیرش حقوق کدهای متن‌باز را دارد و به این ترتیب این بنیاد می‌تواند مدافعانه بخش عمدات از کارهایی باشد که مجوز آنها را در اختیار دارد.

برخی ارائه‌دهندگان که از اختصاص دادن حق خود به هر فرد دیگری خودداری می‌کنند. مدیریت و نگهداشتن حساب مالکیت حقوق کدها، یکی از دردسرهای عمدات هر پروژه متن‌باز است.

۳-۵-۱- مجوزها - متن‌باز و غیر متن‌باز

پس از درک OSD، دولطه بودن و چندین مورد مجوز دوگانه مرتبط با نرم‌افزار، اکنون می‌توانیم نگاهی بر چندین مجوز واقعی داشته باشیم. این قسمت با مجوزهای متن‌باز آغاز می‌شود. در ادامه دو مجوز بسیار مهم GPL و LGPL، به طور مبسوط، شرح داده می‌شوند.

۳-۵-۲- مجوزهای متن‌باز

جدول ۱-۳ برخی از مشهورترین مجوزهایی را که مورد تأیید OSI هستند، شرح می‌دهد. شما و سایر افراد فنی شما می‌توانند با عضو شدن در لیستهای خبری مربوطه، از جدیدترین مجوزهای مورد تأیید OSI مطلع شوند. جالب است بدانید که متن مجوزها توسط مجوز متن‌باز خاصی کنترل نمی‌شود و اگر متن مجوزی قابل تغییر باشد آنگاه لزومی ندارد با مجوز تطابق داشته باشد.

جدول ۱-۳ مجوزهای مشهور متن باز

مجوز	کپی لفت؟	توضیحات
مجوز نرم افزار آپاچی	خیر	این مجوز مورد استفاده کارساز وب آپاچی می باشد. این مجوز شبیه MIT و BSD است. تفاوت اصلی آن اینجاست که باید ذکر شود که نام تجاری آپاچی به همراه نرم افزار مجوزدهی نمی شود. یعنی اگر شما کد منبع آپاچی را تغییر دهید اجازه ندارید برنامه حاصل را "آپاچی" بنامید.
مجوز هنری	خیر	این مجوز برای زبان برنامه نویسی ^۱ طراحی شده است. این مجوز در کل مبهم است و موارد متناقض زیادی دارد. این تناقضها باعث شده است اغلب از آن پرهیز شود و بجائی آن از مجوزهای GPL و MIT استفاده شود.
BSD	خیر	این مجوز از آنجایی که اجازه تغییر و توزیع برنامه بدون انتشار کد منبع را می دهد، بسیار طرفدار دارد. مجوز BSD اجازه می دهد که کد منبع را محترمانه نگه دارید و سپس آن را تحت یک مجوز اختصاصی منتشر کنید. گرچه این مجوز بسیار جذاب به نظر می رسد، اما خواص تشویق کننده کار گروهی موجود در GPL را ندارد. در نسخه قیمتی این مجوز باید هر بار که از این مجوز استفاده می شد، نام دانشگاه کالیفرنیا هم آورده می شد. این بند تبلیغ کننده، باعث می شود با GPL سازگار نباشد که در نسخه های کنونی حذف شده است.
مجوز کلی عمومی (GPL)	بلی	به بحث مربوط به GPL که در ادامه آمده است مراجعه کنید. GPL مجوز هسته لینوکس است. اغلب پروژه های نرم افزارهای آزاد از این مجوز استفاده می کنند.

¹ Perl

مجوز	کپی لفت؟	توضیحات
مجوز کلی عمومی کوچکتر گنو (LGPL)	بلی	به بحث مربوط به LGPL که در ادامه آمده است مراجعه کنید. مشابه GPL است، اما اجازه می‌دهد نرمافزارهای غیرآزاد را به برنامه خود پیوند دهید. همچنان لازم است تغییرات انجام شده در صورت توزیع، منتشر شود.
مجوز عمومی IBM	بلی	یک مجوز تجاری کپی لفت مشابه مجوز عمومی موزیلا که تمہیدات بیشتری برای جبران خسارت درنظر گرفته است. اگر کسی نسبت به کد اصلی یا کدهای مشتق شده ادعایی مطرح کند، مسئولیت آن فقط به گردن مشتق کننده کد است و کسانی که در این بین بهبودهایی روی برنامه مشتق شده یا اصلی داده‌اند متهم نخواهند شد.
مجوز عمومی موزیلا (MPL)	بلی	موزیلا نسخه متن باز مرورگر وب Netscape (که اکنون جزو AOL شده است) می‌باشد. از مشتقات NPL (مجوز عمومی MPL) می‌باشد. برای اطلاعات بیشتر در زمینه NPL به جدول ۲-۳ مراجعه کنید. توسط Netscape طراحی شد تا برخی از محدودیتهای NPL را برچیند. MPL اولین مجوز تجاری متن باز است و اکثر مجوزهای از این دست از این مجوز مشتق شده‌اند.
مجوز MIT	خیر	مجوزی است بسیار ساده که هیچ محدودیتی در استفاده از کد منبع ندارد. تنها لازمه آن، افزودن متن مجوز به تمامی نسخه‌ها می‌باشد. اگر قصد دارید مالکیت کد خود را حفظ کنید و در عین حال هرگونه استفاده از آن را نیز آزاد گذارید، این مجوز انتخاب خوبی است.
مجوز Python	خیر	یک زبان برنامه‌نویسی شی‌گرای تفسیری می‌باشد که در بین جوامع لینوکس و یونیکس از محبوبیت خاصی برخوردار است.

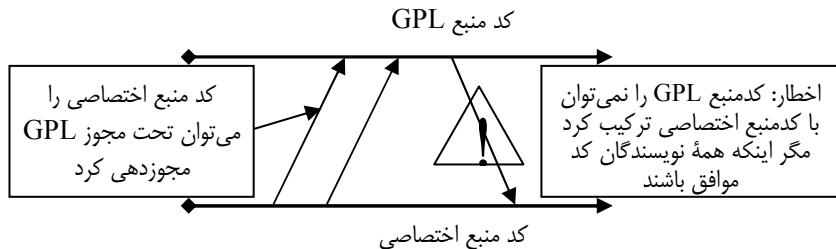
مجوز	کپی لفت؟	توضیحات
مجوز	بلی	Python مخصوص Python است ولی بسیار آزاد است. کافیست نام مجوز Python را در کارهای مشتق شده خود ذکر کنید.
مجوز عمومی Qt (QPL)	بلی	Qt یکی از کتابخانه‌های مرکزی میزکار است. عملکرد آن مشابه Motif است. این مجوز ملزم می‌کند که هر نرم‌افزاری که به آن پیوند می‌خورد کد منبع خود را در دسترس قرار دهد. QPL مشابه LGPL است با این تفاوت که دومی اجازه می‌دهد کدهای اختصاصی با کتابخانه‌ها پیوند برقرار کنند.
مجوز منبع استاندارد SUN (SISSL)	خیر	شرکت Sun این مجوز را در پروژه OpenOffice برای بخش‌هایی که از قالب OpenOffice پروندهای XML و توابع داخلی استفاده می‌کنند، اعمال کرده است. این مجوز به این منظور تعبیه شد تا این دو استاندارد همواره باز بمانند. این مجوز همچنین تضمین می‌کند که تمامی تغییرات اعمال شده به آنها در اختیار عموم قرار گیرد. از این گونه مجوزها زمانی استفاده می‌شود که در عین توسعه به روی روش متن باز نیاز به نوعی کنترل بر روی استانداردهایی که این توسعه در برنامی گیرد احساس شود.

هرچند این جدول مشهورترین مجوزهای متن‌باز را لیست کرده، اما لازم است که نگاه جامع-تری به مجوزهای اصلی کپی‌لفت مثل GPL و LGPL داشته باشیم.

۳-۵-۲- GPL و GPL

مشهورترین مجوز متن‌باز GPL یا GNU General Public License (به معنای مجوز عمومی کلی گنو) است. GPL توسط ریچارد استالمن به عنوان راهی برای جلوگیری از سوء استفاده از مفاهیم نرم‌افزار آزاد، تأثیل شده است (برای مطالعه متن کامل این مجوز به ضمیمه ۳ مراجعه کنید). وقتی نرم‌افزار آزاد و در اختیار عموم باشد، به راحتی می‌توان آن را در اختیار گرفت و آن را به صورت اختصاصی (انحصاری) مورد استفاده قرار داد. استالمن که به این موضوع وقف بود قصد داشت کاری کند که "بده بستان" این‌گونه نرم‌افزارها یا همان دوطرفه بودن ضمانت شود. اگر

شما از نرمافزار متن بازی بهره می‌برید، تغییراتی یا اضافات شما به آن باید به سایرین نیز سود رساند.



شکل ۲-۳: ترکیب کد منبع با مجوز GPL و اختصاصی

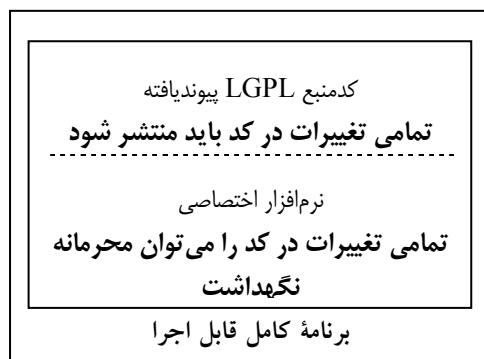
GPL تضمین می‌کند که زنجیره آزادی نرمافزار هرگز شکسته نخواهد شد. این مجوز ملزم می‌کند که هرگونه تغییری که در کد منبع GPL ایجاد می‌کنید و آنچه توزیع می‌کنید باید در اختیار جامعه متن باز قرار گیرد. اولین مجوز کپی لفت بود. شکل ۲-۳ دو نمونه از "درخواره کد منبع" را نشان می‌دهد: یکی GPL و دیگری اختصاصی. این شکل همچنین ملاحظات مربوط به روند تغییر کد منبع را نیز نشان می‌دهد.

متاسفانه بعضی افراد به اشتباه تصویر می‌کنند که منظور از GPL این است که هر چیزی که متأثر از اجرای کد GPL است نیز باید GPL باشد. این تصویر کاملاً غلط است و بهترین مثال از این حالت سامانه عامل لینوکس است. هسته لینوکس تحت GPL مجوزدهی شده است. امروزه برنامه‌های کاربردی بسیاری هستند (مانند پایگاه داده اوراکل) که تحت لینوکس اجرا می‌شوند، ولی GPL نیستند. بخش سوم این کتاب، به جزئیات نوشتن نرمافزارهایی که در ارتباط نزدیک با GPL هستند ولی GPL نیستند، می‌پردازد.

مجوز دیگری که از خانواده خلیلی نزدیک GPL است، LGPL یا Lesser GNU General Public License (به معنای GPL کوچک) است. LGPL برای حل مشکلات استفاده از کتابخانه‌های طراحی شد که تحت GPL مجوزدهی شده بودند. هر برنامه تجاری را که از کتابخانه‌های مشترک^۱ استفاده می‌کند، ملزم به داشتن مجوز GPL می‌کند. ولی LGPL چنین شرطی نمی‌گذارد. مثلاً در محیط لینوکس برنامه‌های کاربردی زیادی هستند که به کتابخانه اصلی زمان اجرای [زبان C در] سامانه، یعنی glibc وابسته‌اند. با مجوزدهی glibc تحت GPL برنامه‌های کاربردی تجاری می‌توانند بدون نگرانی از اینکه مجبور به مجوزدهی تحت GPL شوند، با این کتابخانه پیوند برقرار کنند و از امکانات آن بهره‌مند شوند. وقتی که قصد دارید فناوری طراحی کنید که متن باز باشد و در عین حال می‌خواهید نرمافزارهای تجاری نیز قادر به استفاده از آن فناوری باشند، LGPL را حل خوبی است. ولی دقت داشته باشید هنگام تغییر دادن

¹ Shared

کد منبع LGPL قوانین معمول GPL حاکم هستند؛ تغییرات و پیشرفتها در خود کتابخانه، هنگام توزیع، باید تحت LGPL مجوزدهی شوند. از آن جایی که LGPL اغلب به کتابخانه‌ها اعمال می‌شود، از آن گاهی به GNU General Public License Library نیز یاد می‌شود. شکل ۳-۳ نحوه اتصال یک نرمافزار اختصاصی به نرمافزار GPL و چگونگی ترکیب آن دو برای بوجود آوردن یک برنامه کاربردی کامل را نشان می‌دهد.



شکل ۳-۳: برقراری پیوند بین نرمافزار اختصاصی و کد با مجوز LGPL

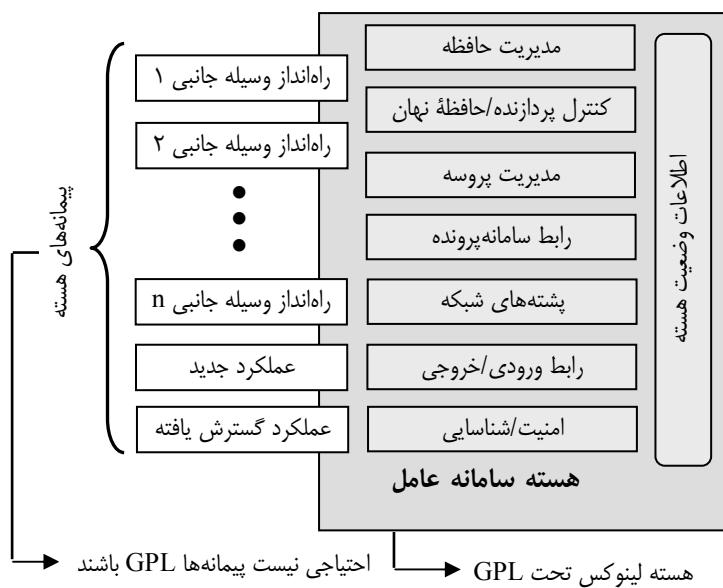
اگرچه امروزه از GPL در طیف وسیعی از نرمافزارها استفاده می‌شود، ولی باعث ایجاد سردرگمیهایی نیز شده است. به همین دلیل ریجارد استالمان اکنون علیه استفاده از آن و در عوض افزودن یک تبصره به GPL را توصیه می‌کند:

"به عنوان یک استثناء خاص، اگر کتابخانه‌ای را با دیگر پرونده‌ها جهت تولید یک برنامه قابل اجرا لینک کنید، این کتابخانه به خودی خود باعث نمی‌شود که برنامه اجرایی حاصل تحت پوشش GPL قرار گیرد. به هر حال، این استثناء موجب نقض دیگر دلایل اینکه چرا پرونده اجرایی ممکن است تحت پوشش GPL قرار گیرد، نخواهد شد." [قرار است این تغییر به همراه تغییرات دیگری در نسخه جدید مجوز GPL (نسخه ۳/۰۰) لحاظ شود]

گرچه GPL و LGPL احتمالاً شناخته شده‌ترین و پرکاربردترین مجوزهای متن باز هستند، مجوزهای بسیار دیگری از این دست وجود دارند. بعضی از این مجوزها قبل از به صحنه آمدن OSD وجود داشتند. برخی از آنها طوری طراحی شده‌اند که محدودیتهای کمتری نسبت به GPL داشته باشند. از آنجایی که تقریباً بیش از ۳۰ مجوز ثبت شده وجود دارد و طراحی موارد جدید عموماً فقط گیج کننده خواهد بود. بنابراین بهتر است تا جایی که ممکن است از مجوزهای موجود استفاده کنید و از طراحی موارد جدید خودداری کنید.

۳-۵-۳- GPL و پیمانه‌های هسته لینوکس

همان‌طور که در فصل ۲ بیان شد، یکی از نقاط قوت نسخه ۲ هسته لینوکس پشتیبانی از پیمانه‌ها در هسته است. پیمانه‌های هسته باعث می‌شوند که بتوان هسته را که به مؤلفه‌های تابعی تقسیم کرد. به جای اینکه مجبور به ترجمه کردن راهاندازهای دستگاه‌ها و عملکردهای جدید به طور مستقیم در هسته باشیم، پیمانه‌ها را می‌توان به محض نیاز، در زمان اجرا به هسته وصله کرد. وقتی یک پیمانه در هسته بارگذاری (load) می‌شود، بخشی از هسته می‌شود. شکل ۳-۴ مشابه یکی از تصویرهای فصل ۲ است، با این تفاوت که ماهیت ذاتی متن‌باز بودن پیمانه‌های هسته را نیز مد نظر دارد.



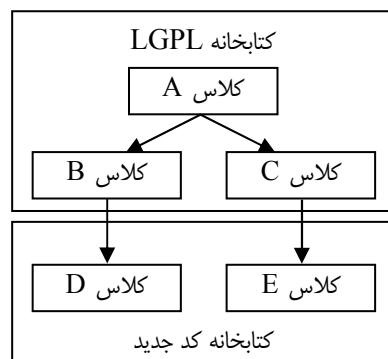
شکل ۳-۴: پیمانه‌های هسته لینوکس و GPL

از نظر مجوز GPL، در اینجا یک مشکل پیش می‌آید. آیا GPL پیمانه‌ها را هم محدود می‌کند؟ از یک طرف پیمانه بخشی از هسته است و باید طبق قواعد GPL محدود شود و از طرف دیگر می‌توان گفت پیمانه کد مجزایی است و از قوانین GPL مستثنی است. این ابهام تبدیل به یکی از بحثهای داغ در بین جامعه متن‌باز شده است. لینوس توروالدز این مسئله را به طریق روشن کرده است. او معتقد است که در صورتی که پیمانه‌های هسته، خود را به واسطه‌های استاندارد محدود کنند، مشمول محدودیتهای GPL هسته لینوکس نمی‌شوند. پیمانه‌های کامپایل شده نمی‌توانند فراخوانهای سامانه جدیدی (System call) (توابعی که برنامه‌نویسان برای ارتباط با هسته از آنها استفاده می‌نمایند) تولید کنند یا قسمتهای موجود هسته را بازنویسی کنند. اگر چه به نظر می‌رسد این استدلال بتواند راه گریزی از GPL باشد، ولی به کارگیری آن عاقلانه به نظر

نمی‌رسد. چرا که توروالدز تنها مالک حق نشر هسته نیست. این موضوع تا حدی پذیرفته شده است که اگر توسعه‌دهندگان هسته اقدام به توسعه کارآیی هسته از طریق به کارگیری استدلال "راه گریز پیمانه‌های هسته" کنند، این راه گریز بسته خواهد شد مثلاً بواسطه یک تجدید نظر ساده در GPL. بنابراین این راه گریز را برای تغییر هسته لینوکس به کار نبرید.

۴-۵-۳ - LGPL و وراثت کلاس

LGPL مجوزی است که اغلب درمورد کتابخانه‌ها استفاده می‌شود. این مجوز به کدهای اختصاصی امکان استفاده از کارایی‌های کتابخانه‌ها را بدون ملزم بودن به رعایت قوانین GPL می‌دهد. مشکل دیگری در ارتباط با توسعه کلاس کتابخانه‌ها وجود دارد. همان طور که در شکل ۴-۳ قابل مشاهده است، زبانهای شی‌گرایی مثل C++ بر پایه سلسه مراتبی از کلاسها ساخته شده‌اند. هر توسعه‌ای که روی سلسه مراتب صورت می‌گیرد، کارآیی جدیدی به کلاس پایه اضافه می‌کند. گرچه قوانین LGPL موقع استفاده از کارایی‌های یک کتابخانه کاملاً واضح هستند ولی این قوانین در توسعه یک کتابخانه محتوی کلاس^۱ خیلی روشن و واضح نیستند. در زمان نوشتن این کتاب بحث روی این موضوع در حال انجام است. از آنجایی که تاکنون جواب واضحی پیدا نشده است، هنگام توسعه کارایی یک Class Library موجود نباید به کار گرفته شود. البته این برای وقتی است که کد شما اختصاصی است.



آیا کتابخانه کد جدید هم باید LGPL باشد؟

شکل ۴-۳: LGPL و ارث بری کلاسها

¹ Class Library

۵-۵-۳- مجوزهای غیر متن باز

تعداد اندکی مجوز غیرمتن باز یا مجوزهایی که منطبق با شرایط OSD نیستند، وجود دارد که ملزمات را ارضاء می‌کنند. به خاطر داشته باشید که صرف اینکه کد منبع در دسترس شما است به این معنا نیست که مجوز آن شرایط OSD را برآورده می‌کند. این مجوزهای غیر متن باز عموماً برای حل موارد خاص تجاری طراحی شده‌اند. در موارد بسیاری ممکن است شرکت حق مجوزدهی کد را نداشته باشد، ولی بخواهد از توسعه اشتراکی و همگانی بهره ببرد. در موارد دیگری هم ممکن است در دسترس گذاشتن کد منبع به منظور کمک گرفتن از توسعه-دهندگان در اسکال زدایی برنامه‌های کاربردی صورت گیرد. در اینجا امکان لیست کردن فهرست جامعی از مجوزهای غیر متن باز وجود ندارد. اکثر مجوزهایی که امکان دسترسی به کد منبع را فراهم می‌کنند ولی OSD را ارضاء نمی‌کنند، در دسته‌بندی زیر قرار می‌گیرند:

۱) مالک می‌خواهد امکان اسکال زدایی یک برنامه کاربردی متکی به کتابخانه را بدهد.

مثلاً مایکروسافت کد منبع را با اکثر کلاس‌های پایه خود (MFC) توزیع می‌کند. این مسئله امکان ریابی برنامه کاربردی را از طریق کد مایکروسافت به برنامه‌نویس می‌دهد. به هرحال، مایکروسافت تا این زمان صاحب یک مجوز متن باز واقعی نشده است.

۲) افراد یا شرکتها از حقوق کامل خود برخوردار نیستند. در موارد بسیاری اشخاص مایلند از مجوزهای متن باز استفاده کنند، ولی به واسطه قراردادهایی که با اشخاص ثالث بسته‌اند، محدود شده‌اند و نمی‌توانند چنین کاری انجام دهنند. در چنین مواردی اغلب مجوز جدیدی خاص آن شرکت یا محصول طراحی می‌شود.

۳) افراد یا شرکتها تمایل دارند بدون در اختیار گذاشتن مالکیتهای معنوی^۱ خود، از یک انجمن برنامه‌نویسان بازخورد دریافت کنند. حالتایی وجود دارد که شرکت می‌خواهد دسترسی به کد منبع وجود داشته باشد تا از مزایای بازخوردها و نظرات انجمنهای توسعه‌دهنده بهره گیرد. هرگونه تغییرات و حقوق نرم‌افزار باید به شرکت برگردد. مجوز منبع مشترک شرکت مایکروسافت در این رده قرار می‌گیرد. این گونه مجوزها مشکل بزرگی در جامعه توسعه متن باز بوجود آورده‌اند. به هرحال ممکن است برای شما مناسب باشند و این در صورتی است که مدل تجاری شما کاملاً وابسته به درآمد ناشی از مجوز نرم‌افزارها باشد و استفاده از مدل تجاری دیگری برای شما مقدور نباشد.

جدول ۲-۳ حاوی لیستی از مجوزهای غیر متن باز مهم است که آشنایی با آنها مفید به نظر می‌رسد. این مجوزها ملزمات OSD را برآورده نمی‌کنند.

¹ Intellectual Propertey

جدول ۲-۳ مجوزهای مهم غیر متن باز

نام مجوز	توضیحات
منع مشترک مايكروسافت	اين مجوز اجازه دسترسى به کد منبع را مى دهد، ولی شما را ملزم به بازگرداندن تغييرات به مايكروسافت مى کند. تمامی حقوق اين تغييرات متعلق به مايكروسافت خواهد بود. در اين مجوز اجازه توزيع کد وجود ندارد.
مجوز عمومي NPL	NPL مختص Netscape است. اين مجوز را برای خاتمه دادن به کشمکشهاي آن شرکت و شرکتهای ثالث ساخت. تا وقتی که موزيلا، کد متن باز واقعی اين مرورگر را فراهم کرد، NPL راهی پيش روی Netscape قرار داد تا بتواند کد را عرضه کند و از پشتيبانی جامعه برنامه نويسان بهره گيرد و در عين حال لطمهای به قراردادهايش با شرکتهای ثالث وارد نگردد.
مجوز منبع جامعه Sun	اين مجوز هم نوعی مجوز کد منبع اشتراکی است. می توان برای بهبود يك پروژه يا تكنولوجی برای آن کد نوشت و در اختیار آن قرار داد، اما همه کارهای عرضه شده باید از استانداردهای شرکت Sun تبعیت کنند. توزیع مجدد آن مستلزم پرداخت هزینه به شرکت Sun Microsystems است. همان گونه که در جدول قبلی اشاره شد، مجوز دیگری نیز دارد که در قالب OSD می گنجد.

این مجوزها آن قدر مهم هستند که باید آنها را مدنظر داشته باشید. به خاطر داشته باشید که مجوزهای جدید مرتباً ساخته می شوند و باید دقت کنید و به روشی بدانید که یک مجوز یا همه شرایط OSD را برآورده می کند یا یک مجوز غیر متن باز است (مورد تایید متن باز نمی باشد).

۳-۶- صادر کردن و رمزنگاری

نرم افزاری که متن باز است، باید از قوانین مربوط به کنترل صادرات نیز پیروی کند. این مسئله خصوصاً برای نرم افزارهایی که قابلیتهای رمزنگاری دارند از اهمیت بیشتری برخوردار است، چرا که ایالات متحده آمریکا و چندین کشور دیگر روی صادرات این گونه نرم افزارها محدودیتهایی اعمال کرده اند.

از آنجایی که نرمافزارهای متن‌باز در سراسر جهان به سادگی نسخه‌برداری و توزیع می‌شوند، هیچ مجوز متن‌بازی نمی‌تواند صریحاً از قوانین ایالات متحده تعییت کند و آن را قید کند. چنین مجوزی با شرط پنجم OSD که بر عدم تعییض بین افراد و گروه‌ها تاکید دارد، در تناقض است. یک مجوز متن‌باز نمی‌تواند عذری در برابر قانون بیاورد. هر کسی که نرمافزاری را می‌نویسد و آن را توزیع می‌کند، با توجه به اینکه از قوانین کنترل صادرات آگاه است، در برابر آن مسئول است و باید از آن قوانین پیروی کند.

این مسأله توزیعهای متن‌باز لینوکس را با مشکل روپرور کرد. اوایل تتها راه ارائه کامل قابلیتهای رمزگاری در لینوکس، انجام آن در خارج از ایالات متحده بود. برای مدتی نرمافزارهای خاصی در این زمینه (مانند SSL و کارسازهای وب خاص) باید از طریق منابع و وب‌گاه‌های خارج از ایالات متحده توسعه می‌یافتند و دریافت می‌شدند.

تفییرات اخیر در قوانین صادرات ایالات متحده به نرمافزارهای متن‌بازی که شامل کدهای رمزگاری می‌شوند، امکان صادر شدن را می‌دهد. با این روش، نرمافزارهای رمزگاری می‌توانند متن‌باز باشند و در عین حال از محدودیتهای قوانین نیز در امان بمانند. توضیحات بیشتری در این زمینه را می‌توانید در وب‌گاه www.bis.doc.gov ببینید.

از آنجایی که همیشه احتمال تغییر در محدودیتها و قوانین وجود دارد، خیلی از نرمافزارهای رمزگاری خارج از ایالات متحده توسعه داده و نگهداری می‌شوند. بنابراین هر تغییری در قوانین صادرات این کشور تاثیر خفیی بر لینوکس و سایر برنامه‌های کاربردی متن‌باز دارد.

۷-۳- متدلوژی توسعه متن‌باز

اگر چه بحث مدل‌های مجوزدهی معمولاً وقتی به متن‌باز می‌رسد، بسیار جالب توجه می‌شود، اما چیزی که واقعاً جالب است، متدلوژی توسعه متن‌باز است. زیبایی واقعی متن‌باز مجوزهای آن نیست، بلکه فرآیند توسعه آن است.

توصیف کلی متدلوژی توسعه متن‌باز، تعداد زیادی از پروژه‌ها را پوشش می‌دهد. اما خصوصیات و ویژگیهای خاص آن بسته به اندازه و کاربرد پروژه متغیر است. مقاله (و کتاب) "اریک ریموند" جزئیات کار وی را شرح می‌دهد. ریموند متغیر بود که چگونه فرآیندی که به نظر می‌رسد هر قانون شناخته شده‌ای از توسعه سنتی نرمافزار (مدل کاترال) را نقص می‌کند، هنوز جواب می‌دهد. مقاله ریموند مدل بازار را که پیشنهاد وی بود، شرح می‌دهد. در زیر عناصر کلیدی این متدلوژی بدون اعمال پروژه آورده شده است. شما باید واژگان، ابزارها، نقش آفرینان و فرآیند آن را بشناسید.

۷-۴- مجوز

گرچه فرآیند توسعه مستقیماً در گیر مسائل مربوط به مجوز نیست، این مجوز است که محدودیتها و مرزهایی را که به واسطه آنها فرآیند توسعه شکل می‌گیرد، تعیین می‌کند. این شرایط زیربنای توسعه همگانی و اشتراکی را تضمین می‌کنند. از آنجایی که GPL و LGPL و دیگر مجوزهایی

که ملزمات دوطرفه بودن را رعایت می‌کنند، بر این اصل استوارند که تغییرات و اصلاحات باید به انجمان برگشت داده شود، بنیان و زیربنای توسعه همگانی تضمین می‌شود.

۲-۷-۳- روشهای

روش توسعه متن باز را می‌توان روش توسعه لینوکس نیز نامید. لینوس توروالدز یک نمونه برای نشان دادن این موضوع است که به روشن بازار کار می‌کند و خوب هم کار می‌کند. این روش به صورتهای زیر قابل تصور است:

۱. شرکت

در این روش شرکت رهبری یک پروژه را بر عهده می‌گیرد. این فرآیند معمولاً به این ترتیب است که تعدادی توسعه‌دهنده را به پروژه اختصاص داده می‌شود و با سرمایه‌گذاری زیرساخت پروژه شکل می‌گیرد و برای نگهداری پروژه اقدام می‌شود. حتی در این روش هم، شرکت یک برنامه‌نویس را به عنوان نگهدارنده اصلی انتخاب می‌کند. SUN، IBM، HP و شرکتهای بزرگ دیگری پروژه‌های متن باز را با این روش هدایت و رهبری می‌کنند.

۲. بنیاد

بنیادها اغلب برای سرمایه‌گذاری روی پروژه‌های بزرگ ایجاد می‌شوند. شرکتها و افرادی که مایلند به موقفيت این پروژه کمک کنند، می‌توانند از طریق این سازمان غیرانتفاعی، پروژه را یاری کنند. مواردی وجود دارد که یک شرکت رهبری یک پروژه را بر عهده گرفته است و در عین حال بنیادی هم برای آن بنا کرده است. این معمولاً به این خاطر است که شرکتی که رهبری پروژه را بر عهده دارد، برای سرمایه‌گذاری روی آن از منابع مالی کافی برخوردار نیست. بنیاد گنوم (GNOME)، مثال خوبی از این روش توسعه است. شرکتهای زیادی در بنیاد گنوم سرمایه‌گذاری کرده‌اند. مثلًا Ximian، [و هم اکنون شرکت Novell] مؤسسه‌ای است که رهبری پروژه میزکار گنوم را تا نهایی کردن بر عهده گرفته است.

۳. کمیته

از روش کمیته نیز برای پروژه‌های بزرگ استفاده می‌شود. کمیته‌ها معمولاً میدانی برای تصمیم‌گیری فراهم می‌کنند. این روش، رویکرد تصمیم‌گیری جمعی را (در مقابل رویکرد معمولی شامل یک نگهدارنده) ارائه می‌کند. مترجمهای GCC که اغلب در لینوکس استفاده می‌شوند به این روش مدیریت می‌شوند.

۴. افراد

پروژه‌های کوچک معمولاً توسط افراد اجرا می‌شود. در واقع هزاران نوع از این پروژه‌ها

وجود دارد. اکثر این گونه پروژه‌ها روی وب‌گاه^۱ Source Forge قرار دارند و اخیراً Savannah هم که توسط پروژه گنو ارائه شده است، به مخزن‌های نرم‌افزار آزاد اضافه شده است. این گونه پروژه‌ها اغلب کمتر از پنج برنامه‌نویس دارند. بیشتر این پروژه‌ها این گونه آغاز می‌شوند که یک برنامه‌نویس در بی‌یافتن راه حلی برای مشکل خود است و شروع به برنامه‌نویسی می‌کند تا نیاز خود را برآورده کند و آن را متن باز می‌کند.

۵. هسته لینوکس

لینوکس توروالدز از این روش خاص برای توسعه بزرگترین پروژه متن‌باز (هسته لینوکس) استفاده کرد. توروالدز این مدل را در طی سالها تکمیل کرده است. این روش مبتنی بر سلسه مراتبی از اطمینان در انجمن توسعه هسته است. در بخش ۳ این کتاب شرح داده خواهد شد که چگونه می‌توان مدل نظری فرآیند توسعه هسته لینوکس را در یک مؤسسه به کار گرفت.

۶. گنو

این روش توسعه خیلی جا افتاده نیست و اغلب ترجیح داده می‌شود که از روش‌های قبلی استفاده شود. روش گنو از متداول‌یهای توسعه قراردادی (که بعداً در مورد آن، تحت عنوان مدل توسعه کاتدرال، خواهید آموخت) برای تولید نرم‌افزار آزاد استفاده می‌کند.

۷-۳-۳-۳. صفات

صفات، مجموعه‌ای از خصوصیات هستند که در اغلب پروژه‌های متن‌باز مشترک‌کند. این مجموعه مشخصات پایه مهمی برای شکل دادن رفتار جامعه برنامه‌نویسان هستند. اغلب پروژه‌های متن‌باز صفات زیر را به طور مشترک دارند:

۱. نگهدارنده

اکثر پروژه‌ها توسط یک نگهدارنده هدایت می‌شوند. نگهدارنده فردی است که پروژه را بنیان می‌نهد یا کسی است که توسط نگهدارنده قبلی برای هدایت پروژه معین می‌شود. نگهدارنده کارهای ارائه شده را رد یا قبول می‌کند، وصله‌ها را به کار می‌گیرند، کاستی را کنترل و مدیریت می‌کنند و در صورت لزوم با دیگر پروژه‌ها تعامل برقرار می‌کند. وقتی یک نگهدارنده دیگر علاقه‌ای به ادامه هدایت پروژه نداشته باشد یا وقت این کار را نداشته باشد، کار خود را به نگهدارنده جدیدی می‌سپرد. نگهدارنده جدید از بین افرادی انتخاب می‌شود که در پروژه‌ها شرکت فعال دارند و داوطلب این سمت

¹ <http://www.sf.net>

هستند. مواردی هم وجود دارد که یک مؤسس کمیته‌ای برای پروژه تأسیس کرده است. همچنین در مواردی که هدایت پروژه بر عهده شرکت است، ممکن است گروهی از کارمندان شرکت برای نگهداری آن انتخاب شوند.

۲. احترام

در طول زمان، احترام به افراد در جامعه برنامه‌نویسان به صورت یک شاخص در آمده است. اگر توسعه‌دهنده‌ای کم کاری کند، از سوی دیگر اعضاء مورد تذکر قرار می‌گیرد. انتقاد و تحسین اعضاء در صورت لزوم اعمال می‌شود. افرادی که فعال هستند، از احترام خاصی در این جامعه برخوردار هستند. افرادی مثل توروالدز جایگاه خاصی در انجمن دارند. شرکت‌کنندگان در یک پروژه قبل از اینکه پذیرفته شوند به دقت مورد بررسی قرار می‌گیرند و کسانی که نتوانند همکاری فعال داشته باشند رد می‌شوند.

۳. افراد به عنوان نگهدارنده

حتی اگر شرکتها پروژه‌های متن‌باز را مدیریت کنند، نگهدارنده اغلب یک فرد است. اگر نگهدارنده شرکت خود را تغییر دهد، ممکن است همچنان به عنوان نگهدارنده رسمی کد باقی بماند. در بعضی موارد براساس تغییر اولویتها در شرکتها تغییراتی رخ می‌دهد. در چنین شرایطی نگهدارندها مشعل را به دست دیگری می‌سپارند. مردم توروالدز را به عنوان نگهدارنده هسته لینوکس می‌شناسند نه Transmeta (شرکتی که او در آن کار می‌کند) را.

۴. ابزارها

همه توسعه‌دهنگان جامعه به زیرساخت کاملاً ساخت‌یافته‌ای همراه با ابزارهای توسعه رایج عادت کرده‌اند. مثلاً CVS به عنوان ابزار مدیریت [نسخه‌های مختلف کد منبع] کاملاً پذیرفته شده است و اغلب برای پیگیری اشکالات به کار می‌رود. قابل ذکر است با اینکه CVS ابزار مدیریتی است که در طیف وسیعی از آن استفاده می‌شود، برای هسته لینوکس به کار نمی‌رود. هسته لینوکس مبتنی بر وصله‌ها است. SourceForge و اکنون Savannah برای نرم‌افزارهای آزاد و متن‌باز زیرساخت توسعه مناسبی ارائه می‌دهند.

۵. عرضه زود هنگام و مکرر^۱

یکی از تفاوت‌های اساسی بین روش توسعه سنتی کاتدرال و روش توسعه بازاری متن‌باز، عرضه کارهای در حال توسعه است که در روش بازار وجود دارد. این حلقه‌های عرضه سریع و مداوم به توسعه‌دهنگان امکان تشخیص نقاط ضعف را در کمترین

¹ Release early and often

زمان می‌دهد که بدین وسیله می‌توان آنها را در فرآیند توسعه از بین برد. توسعه‌دهندگان به تناوب از نسخه‌های عرضه شده زودهنگام در کارهای روزانه خود استفاده می‌کنند تا عیوب آنها را پیدا کنند. این ویژگی دلیلی است برای این سؤال که چرا محصولات متن‌باز نسبت به محصولات سنتی خیلی سریع‌تر و با کیفیت خیلی بالاتر توسعه می‌یابند.

۶. عرضه پروژه به عنوان محصول پس از آماده شدن

همان طور که در فصل دوم گفته شد، بسیاری از شرکتها، نرمافزار را در طول مدت زمانی محدود که بستگی به فاکتورهایی نظیر فشار بازار و رقبا دارد، طراحی می‌کنند که معمولاً نتایج خوبی دربردارد. در فرآیند توسعه متن‌باز عموماً وقتی انجمن اقدام به ارائه محصول می‌کند که نگهدارنده تشخیص دهد محصول آماده است. خیلی از شرکتها وقتی با پروژه‌های متن‌باز روپرتو می‌شوند از فقدان طرح کلی برای کار احساس ناراحتی می‌کنند. به هر حال کسانی که این شرط را می‌پذیرند، از مزایای نرمافزار دارای کیفیت بیشتر، وصله‌های کمتر و یک معماری بهتر برای سامانه، بهره‌مند می‌شوند.

ترکیب این صفات متن‌باز با روشهای مختلف توسعه، یک فرآیند توسعه همگانی موثری را ایجاد می‌کند. جا انداختن این فرهنگ همکاری در شرکت، زمان و شهامت می‌طلبد ولی ارزش آن را دارد. مثلاً لینوکس و کارساز وب آپاچی را در نظر بگیرید به عنوان دو پروژه بزرگ متن‌باز که چنان رشد یافته‌اند که بار مدیریت بخش عظیمی از اینترنت را بر عهده دارند.

۸-۳- جمع بندی

در این فصل با جزئیات متن باز آشنا شدید و نحوه ارتباط آن با نرم افزار آزاد را درک کردید. ما OSD را به طور مفصل شرح دادیم. همچنین به تعدادی از مجوزهایی که شرایط OSD را برآورده می‌کنند، اشاره کردیم. در نهایت به این موضوع پرداختیم که چگونه جامعه، فرآیند توسعه مبتنی بر روش بازار را اجرا می‌کند و بعد از آن ویژگیهای کلیدی پروژه‌ها در جامعه ذکر شد. در بخش سوم بحث خود را در مورد متن باز ادامه می‌دهیم و سعی خواهیم کرد برای سوالات زیر جواب پیدا کیم.

۱. خطرات تجاری مرتبط با GPL و LGPL چیست و چگونه می‌توان آنها را مدیریت کرد؟
۲. متن باز چگونه بر مدل‌های تجاری موجود تاثیر می‌گذارد؟
۳. شرکتها چگونه می‌توانند از روش توسعه بازار بهره گیرند؟
۴. شرکتها چگونه می‌توانند روی پروژه‌های متن باز کار کنند؟
۵. چه فرآیندهای تجاری کلیدی متاثر از توسعه متن باز هستند؟

در فصل بعد تعدادی از انجمنها و سازمانها را در جامعه بزرگ متن باز بررسی می‌کنیم و همچنین نقشی را که هریک بر عهده دارند مورد توجه قرار می‌دهیم.

فصل چهارم انجمنها و شرکتها

متن باز مجموعه‌ای از انجمنها است. در فصل سوم برخی از روش‌های مختلف اجرای پروژه‌ها از روش شرکتی تا بنیادی شرح داده شد. نوع دیگری از سازمانها وجود دارند که برای لینوکس و حرکت متن باز خدمات پشتیبانی ارائه می‌کنند. داشتن درک بهتری از تمام انجمنهای موجود و آشنایی با آنها در درک عظمت این جنبش مؤثر است. همچنین تعاملات پیچیده‌ای بین شرکتهای مختلف وجود دارد که شما تنها در صورتی آنها را درک خواهید کرد که نسبت به این جوامع شناخت داشته باشید.

این فصل طرح کلی از انجمنهای شاخص را ارائه می‌دهد. این بحث را با انجمن هسته لینوکس شروع خواهیم کرد. سپس نگاهی به برخی مؤسسات اصلی متن باز خواهیم انداخت و در نهایت تعدادی از سازمانهای صنعتی را بررسی می‌کنیم.

فهرست کردن لیست همه گروه‌ها، انجمنها، مؤسسات و سازمانهای موجود در این فصل مقدور نیست، ولی به هر حال برای شروع کافی است. در مورد برخی از این سازمانها و انجمنها می‌توان یک فصل کامل یا حتی یک کتاب کامل نوشت. هدفی که دنبال می‌شود این است که یک دید کلی به شما داده شود، نه شناخت عمیق نسبت به همه گروه‌ها و پروژه‌ها. همچنین نگاهی اجمالی به برخی پروژه‌های کلیدی متن باز که برای صاحبان تجاری خود درآمدزا بوده‌اند، خواهیم انداخت. اگر در حال توسعه برنامه‌های کاربردی بازار گانی هستید، مواطن باشید شما نفر بعدی که با تخریب تدریجی بازار خود روبرو خواهد بود، نباشد.

احتیاجی به خواندن کامل این فصل نیست. می‌توانید این فصل را به طور اجمالی بررسی کنید و جزئیات انجمنی را که فکر می‌کنید به کار و تجارت شما مربوط است، با تفصیل بیشتری مطالعه کنید. همچنین می‌توانید نگاهی به انجمنهایی بیاندازید که با آنها آشنایی ندارید. بعدها هنگام نیاز به مراجعة به وب‌گاه یک انجمن خاص، به این فصل مراجعه کنید.

۴-۱-۱- لینوکس

همان‌طور که در فصلهای اول و دوم فرا گرفتید، لینوکس درحقیقت یک هسته است. انجمن هسته شامل گروه‌هایی است که بر خود هسته، امور مختص پردازنده‌ها تمرکز دارند و روی مواردی مانند سامانه‌پرونده و دستگاه گردانها کار می‌کنند.

۴-۱-۱- هسته^۱

kernel.org مرکز دنیای هسته است، گروهی از مهندسین که بر عمیق‌ترین و پیچیده‌ترین جنبه‌های توسعه نرم‌افزار تکیه کرده‌اند و با تمام توان کار می‌کنند. همانند انجمنهای دیگر، این انجمن بر پایه ارتباطات پست الکترونیکی از طریق لیستهای الکترونیکی (به این معنی که نامه

¹ <http://www.kernel.org>

الکترونیکی که به یک آدرس فرستاده می‌شود به طور خودکار به دیگر اعضاء نیز فرستاده می‌شود) عمل می‌کند. با توجه به اینکه ترافیک پست الکترونیک در این گونه لیستها بسیار بالا است و تنها کسانی که عضو گروه توسعه هسته هستند، می‌توانند به این انجمن متصل شوند.

۴-۱-۲- پردازنده‌ها X86 (IA-32)

خانواده پردازنده‌های IA-32 انجمن مجازی ندارد. زیرا در دل تمام انجمنهای توسعه هسته وجود دارد و به عنوان مبنا درنظر گرفته می‌شود.

خانواده پردازنده‌های ایتانیوم^۱ یا IPF (IA-64)

IPF جانشین برای خانواده پردازنده‌های IA-32 IA شناخته می‌شود. این رویدادی یک شبه اتفاق نخواهد افتاد. در آینده‌ای نزدیک IA-32 از نظر قیمت/کارآیی به خصوص در زمینه سامانه‌های کاربری پیشگام خواهد بود. IPF در ابتدا در زمینه محاسبات سنگین پیشرو خواهد بود و در آینده وقتی که برنامه‌های کاربری نیاز به محاسبه ۶۴ بیتی دارند، تبدیل به پردازنده برتر خواهد شد و اکوسیستمی محصولات مبتنی بر کالا حول معماری آن رشد خواهد کرد.

لينوكس تحت IPF، پروژه‌ای است که در ابتدا به رهبری "ديويد موزبرگ"^۲ و "استفان ارانيان"^۳ در آزمایشگاه‌های HP آغاز به کار کرد. اینتل و شرکت‌های دیگری که ملاحظات عدم افشاء اطلاعات محرمانه اینتل را رعایت می‌کردند نیز در این فضا کار می‌کردند. کمی بعد از شروع پروژه گروهی از شرکتها شامل HP، SGI، IBM و چند شرکت دیگر کنسرسیومی با نام گروه تریلیان^۴ تشکیل دادند. تریلیان به شرکتها اجازه می‌داد بدون افشاء ویژگیهای Itanium به عموم، در زمینه پشتیبانی لينوكس تحت آن پردازنده کار کنند. وقتی Intel، Itanium را گسترش داد، گروه تریلیان منحل شد و کد لينوكس تحت IPF به صورت متن‌باز در دسترس همگان قرار گرفت.

شبیه ساز IPF برای خانواده پردازنده IA-32 موجود می‌باشد. یک نکته عجیب در مورد این شبیه-ساز این است که به برنامه‌نویسان این امکان را می‌دهد که از رایانه‌های کیفی^۵ و رومیزی ارزان قیمت خود برای برنامه‌نویسی لينوكس تحت IPF استفاده کنند. اجزای دیگری نیز روی وب سایت رسمی لينوكس IPF موجود است.

اکنون، دیوید به عنوان نگهدارنده هسته لينوكس IPF ایفای نقش می‌کند. او همچنین مثال خوبی از نگهدارنده به عنوان یک فرد (به جای شرکت) است.

¹ <http://www.linuxia64.org>

² David Mosberger

³ Stephone Eranian

⁴ Trillian

⁵ Laptop

Alpha

پردازنده آلفا^۱ از پردازنده‌های مشهور در تاریخ لینوکس است. در واقع آلفا برای مدت زمان طولانی پردازنده برتر برای لینوکس در برنامه‌های کاربردی محاسباتی و با کارآیی بالا بود. برخی ارائه‌دهندگان توزیعها برای پیاده‌سازی آلفا پشتیبانی ارائه داده‌اند، به هر حال Compaq قبل از اینکه به مالکیت HP درآید، قریب‌الوقوع بودن پایان عمر آلفا را اعلام کرده بود و IPF را جایگزین آینده آن شناخته بود. عمدۀ مالکیت معنوی آلفا متعلق به اینتل است.

PowerPC^۲

IBM و Motorola با هم PowerPC را تولید کردند. انجمن PowerPC عمدتاً بر ارائه پشتیبانی لینوکس برای رایانه‌های مکینتاش که ریزپردازنده PowerPC را به کار می‌گیرند، تمرکز دارد. همچنین برخی از سامانه‌های IBM را پشتیبانی می‌کند. برخی توزیعها نیز نسخه‌هایی برای این معماری طراحی کرده‌اند.

PA-RISC^۳

PA-RISC خانواده پردازنده‌ای است که توسط شرکت HP توسعه یافته است. HP سرمایه‌گذاری عظیمی نیز در IPF کرده است و همراه با اینتل پیش می‌رود. این شرکت، در حال توسعۀ PA-RISC برای IPF به عنوان ریزپردازنده راهبردی آینده است. گروه Puffin که یک شرکت کوچک برنامه‌نویسی است، انتقال هستۀ لینوکس به پردازنده PA-RISC را بنیان نهاده است. HP کمک خود را به شکل پشتیبانی توسعه و مستندسازی ارائه داد. گروه Puffin که به کمک LinuxCare شکل گرفت، تاکنون فعالیتهای (برنامه‌نویسی) زیادی برای لینوکس تحت PA-RISC انجام داده است.

۴-۳-۱- سامانه‌های پرونده^۴

لينوکس طيف گسترده‌ای از سامانه‌پرونده را پشتیبانی می‌کند. سامانه‌های پرونده در حالت کلی نحوه ذخیره شدن، ایندکس شدن و بازیابی داده‌ها را از دیسک تعریف می‌کنند. متداول‌ترین سامانه‌پرونده موجود برای لینوکس، سامانه‌پرونده ext2 می‌باشد. اگر یکی از توزیعهای استاندارد لینوکس را در اختیار داشته باشد، این سامانه‌پرونده را در آن پیدا خواهد کرد. به هر حال به دلایلی ممکن است از سامانه‌های پرونده دیگر استفاده شود. برخی موقع ممکن است بخواهید دیسکها را بین سامانه‌های عامل مختلف تسهیم کنید که در این صورت باید از ساختار مشترکی پیروی کنید. ممکن است کارآیی یا قابلیت اطمینانی مدنظر شما باشد که سامانه‌پرونده خاصی را بطلبید. بهتر است برای تصمیم گرفتن در مورد اینکه بهترین سامانه‌پرونده برای شما چه سامانه

¹ <http://www.linuxalpha.org>

² <http://www.linuxppc.org>

³ <http://www.parisc-linux.org>

⁴ File Systems

پرونده می‌تواند باشد، با استفاده از برنامه‌های کاربردی خاص، یکسری آزمون انجام دهد. در زیر چند سامانه‌پرونده مهم آورده شده است:

EXT¹

سامانه‌های پرونده journaling ، درجه بالایی از انعطاف‌پذیری و قابلیت اطمینان برای سامانه‌های عامل فراهم می‌کنند. حین فعالیت عادی سامانه‌عامل، مقادیر زیادی عملیات ذخیره‌سازی و بازیابی دیسک رخ می‌دهد. از آنجایی که این اعمال با سرعت نسبتاً کمی صورت می‌گیرند، همه سامانه‌ها از تکنیکهایی برای سرعت بخشیدن به این اعمال بهره می‌گیرند و معمولاً² این کار با تعییه حافظه‌های موقتی³ در حافظه اصلی سامانه صورت می‌گیرد. به هر حال اگر حین این عمل مشکلی در سامانه پیش آید، ممکن است صحت داده‌های روی دیسک نقض شود. سامانه‌های پرونده journaling بر این مشکل که اغلب به عنوان یکی از ضعفهای سامانه‌های عامل بشمار می‌رود، فائق آمده است.

XFS⁴

هاینس رایز⁵ طراح سامانه‌پرونده ReiserFS است که یک سامانه‌پرونده journaling با کارآیی بالا تحت GPL می‌باشد. در بحث مجوز دوگانه ذکر شد که رایزر مجوز استفاده از سامانه‌پرونده خود را به عرضه‌کنندگان سامانه‌های عامل تجاری داده است. این ویژگی امکان استفاده از فناوری سامانه‌پرونده یکسان و تسهیم حافظه بین لینوکس و گونه‌های دیگر یونیکس را فراهم می‌کند.

۴-۱-۴- پشتیبانی سخت افزار

امکان لیست کردن انجمنهای پشتیبانی کننده همه دستگاه‌های سخت افزاری امکان ندارد. بهترین راه برای اطمینان از پشتیبانی یک توزیع لینوکس از یک دستگاه خاص، بررسی لیست سخت افزارهای پشتیبانی شده توسط آن توزیع است (که در وب‌گاه شرکت توزیع کننده آن موجود است).

در فصل پنجم، توزیعهای لینوکس را با تفصیل بیشتری مورد بررسی قرار خواهیم داد و همچنین در مورد نحوه پشتیبانی توزیعها از دستگاه‌های سخت افزاری بحث خواهیم کرد.

۴-۲- خدمات وب و کارسازهای برنامه‌های کاربردی

قبل‌اً در مورد نقش کارساز وب آپاچی در رشد چشمگیر لینوکس بحث کردیم. ولی آپاچی، تنها یک نرم‌افزار نیست و کار آن منحصر به یک کارساز وب ساده نمی‌شود. وب از ارائه محتوای ایستا مثل نشر و چاپ به ارائه محتوای پویا که بیشتر شامل تعاملات پایگاه داده‌ای می‌باشد، گسترش یافته است. علاوه بر این، شاهد تکامل و گسترش تجارت الکترونیکی، منطق تجارت و

¹ <http://oss.software.ibm.com/developeropensource/ifs>

² Cache

³ <http://www.namesys.com>

⁴ Heniz Reiser

خبرأً خدمات وب نوینی هستیم. انجمنهای متن بازی وجود دارند که پیشنهادهایی برای این فضای در حال رشد و توسعه ارائه می‌دهند. لازم به یادآوری است که همه آنها منحصر به لینوکس نیستند؛ متن باز برای سامانه‌های عامل غیرلینوکس نیز راه حل‌های زیادی ارائه کرده است. در اینجا به برخی از آنها نگاه گذراي خواهيم انداخت:

Apache^۱ آپاچی^۱ به صورت یک بنیاد غیرانتفاعی (نه به قصد به دست آوردن سود مالی) به منظور پیش برد توسعه هسته کارساز وب و همه خدمات پویایی که حول و حوش آن شکل می‌گیرند و توسعه می‌یابند، بنیان گذاری شد.

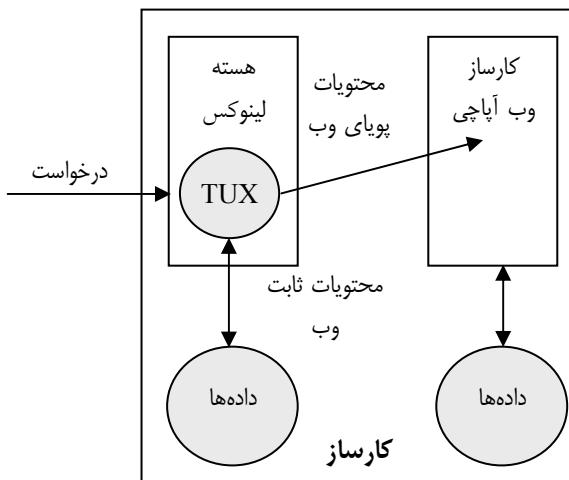
در وب گاه این انجمن تعداد زیادی زیرانجمن می‌یابید که پا به پای فناوریهای جدید و در حال گسترش در زمینه وب و کارسازهای برنامه‌های کاربردی، تکامل یافته‌اند. یکی از این زیرانجمنها، انجمن XML برای آپاچی است. XML قالب داده‌ای مشترکی برای توسعه کارسازهای برنامه‌های کاربردی برای نسل بعدی این کارسازها ارائه می‌دهد.

TUX^۲ TUX یک کارساز وب مبتنی بر هسته است. کارسازهای وب مانند آپاچی مثل یک برنامه کاربردی روی هسته لینوکس اجرا می‌شوند. چیزی که TUX را خیلی ویژه و خاص می‌سازد این است که به صورت جزئی از هسته اجرا می‌شود و در نتیجه می‌تواند از کارآیی بالایی برخوردار باشد و بر کارآیی خدمات لایه‌های بالاتر بیفزاید.

شكل ۱-۴ تصویر ارتباط بین TUX درون هسته و کارساز وب آپاچی (خارج هسته) را نشان می‌دهد. ترکیب TUX بین کارساز وب برای بهبود عملکرد کارسازها بسیار حائز اهمیت است.

¹ <http://www.apache.org>

² <http://www.redhat.com/products/software/linux/tux/>



شکل ۴: استفاده از TUX به همراه کارسازهای وب

JBOSS^۱

JBOSS نسخه حرفه‌ای جاوای ۲ (J2EE) که به صورت متن‌باز پیاده‌سازی شده است می‌باشد. J2EE نوعی فناوری کارسازهای برنامه کاربردی است. پیاده‌سازی‌های تجاری J2EE عبارتند از iPlanet از SUN، BEA از Weblogic و IBM از Websphere. به واسطه رعایت مجوزهای SUN، JBOSS نمی‌تواند به عنوان نسخه تأیید شده J2EE، صاحب مجوز شود. اگرچه از طریق کار با بنیاد آپاچی، SUN در نظر دارد که به تدریج پیاده‌سازی‌های متن‌باز J2EE را آزاد کند.

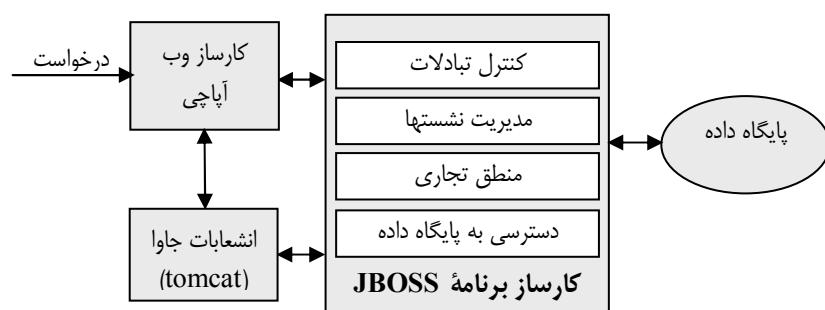
کارسازهای برنامه‌های کاربردی به طور چشمگیری قابلیتهای وب کارسازهای سنتی را افزایش داده‌اند. برخی از این قابلیتها شامل خدمات تراکنش برای اطمینان از کنترل همزمانی، خدمات منطق تجارت برای امکان ساخت سریع برنامه‌های کاربردی با ترکیب اجزاء از پیش ساخته شده و خدمات پایگاه داده‌ای برای دسترسی به داده‌های شرکت، می‌شوند.

با رشد سریع اینترنت، توسعه برنامه‌های کاربردی زیادی مبتنی بر فناوری‌های کارسازهای برنامه‌های کاربردی صورت گرفت. مبنای دنیای کارسازهای برنامه‌های کاربردی یونیکس/لینوکس اغلب فناوری جاوا و مبنای دنیای ویندوز، فناوری مایکروسافت.NET. (دات نت) می‌باشد.

شکل ۲-۴ ارتباط بین یک کارساز وب، کارساز برنامه کاربردی و پایگاه داده را نشان می‌دهد. بسته به پیچیدگی برنامه کاربردی، کارسازهای برنامه کاربردی متن‌باز می‌توانند بستر مطمئنی

^۱ <http://www.jboss.org>

برای پیادهسازی‌های تجاری فراهم کنند. در بخش سوم، هنگام بررسی مدل‌های تجاری، نگاه عمیق‌تری به تأثیر متن‌باز بر مدل‌های تجاری سنتی خواهیم انداخت.



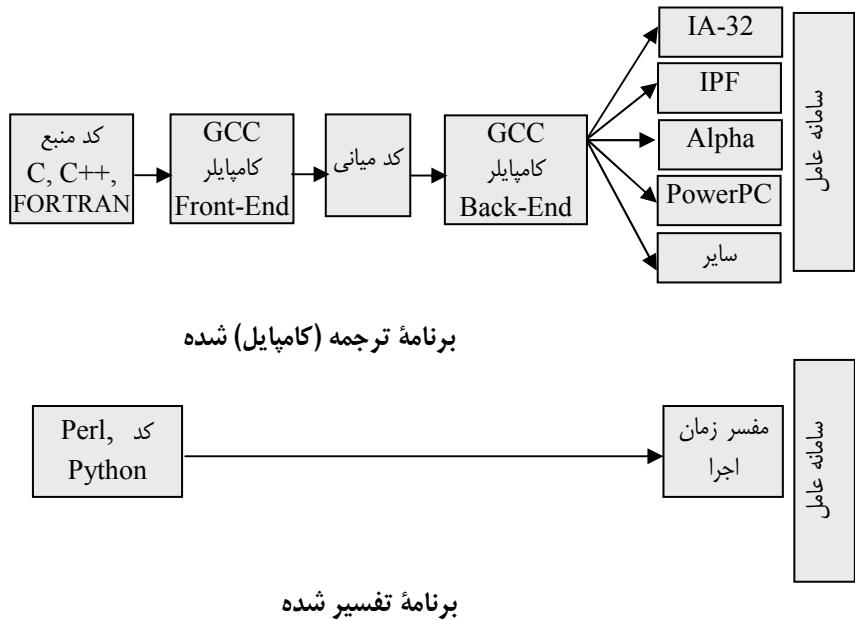
شکل ۴-۲: یک کارساز برنامه کاربردی متصل به کارساز وب و پایگاه داده

۴-۳-۴- زبانهای برنامه‌نویسی

تعدادی زبان برنامه‌نویسی متن‌باز وجود دارد که بعضی از آنها تفسیری (جی‌اس‌اچ‌ای برنامه خط به خط به کد ماشین ترجمه می‌شوند) و بعضی از آنها مترجمی (قبل از اجرا به کد ماشین ترجمه می‌شوند) هستند.

شکل ۳-۴ اختلاف نحوه اجرای زبانهای تفسیری و مترجمی را روی سامانه نشان می‌دهد: قسمت عمده هسته لینوکس - دستگاه گردانها و مؤلفه‌های دیگر هسته - با زبان برنامه‌نویسی C نوشته شده است. به هر حال بیشتر نرم‌افزارهای تحت وب و دست‌نوشته‌ها^۱، مبتنی بر زبانهای تفسیری مثل Perl و Python هستند. در مورد ارتباط دادن این زبانها به طور خاص به لینوکس دقت کنید. این زبانها و مفسر/مترجمهای آنها، روی سامانه‌های عامل متعددی (مثل پیادهسازی‌های مختلف یونیکس و ویندوز) موجود هستند.

¹ Scripts



شکل ۴-۳: برنامه‌های ترجمه شده و تفسیر شده

GCC^۱ احتمالاً GCC، مهمترین ابزاری بود که به لینوس توروالدز امکان نوشتن هسته لینوکس را داد. GCC، مجموعه‌ای از مترجمهایی است که برای ترجمه زبانهای سطح بالا به زبان ماشین به کار می‌روند. فناوری مترجم در علم رایانه خیلی مهم است، چرا که می‌تواند نقش مهمی در کارآیی برنامه کاربردی ایفا کند. مترجمها، عمدهاً به مؤلفه‌های front-end و back-end تقسیم می‌شوند. مؤلفه front-end مترجم، زبانهای سطح بالا مثل C، C++، فرترن یا موارد دیگر را به زبانهای میانی ترجمه می‌کنند. مؤلفه back-end مترجم، کار سنگین اصلی را انجام می‌دهد. این مؤلفه مسئول دریافت کد میانی و انجام دو وظیفه عمده است: (۱) ترجمه به زبان ماشین و (۲) بهینه سازی. ترجمه به زبان ماشین امکان اجرا شدن برنامه کاربردی شما را روی سامانه مقصد را می‌دهد. این ترجمه به شکلی است که برنامه می‌تواند روی معماری ریز پردازنده مقصد اجرا شود و با سامانه‌عامل سازگار باشد. روش‌های بهینه‌سازی کد، موضوع تحقیقات و مطالعات علوم پیشرفته رایانه هستند. در بهینه‌سازی، کد برای اطمینان از اجرا با حداقل سرعت و کارآیی ممکن توسط پردازندۀ مقصد، ساماندهی می‌شود.

^۱ <http://gcc.gnu.org>

GCC قسمت اصلی پروژه گنو است. امروزه از مترجمهای GCC در طیف عظیمی از سامانه‌های عامل استفاده می‌شود. در واقع ISVها (تولیدکنندگان نرم‌افزارها) برای تضمین کردن قابلیت اجرا بر روی بسترهای مختلفی که پشتیبانی می‌کنند، GCC را برای طراحی نرم‌افزار مورد نظر خود انتخاب می‌کنند. هدف اولیه طراحی GCC این بوده است که ابزار برنامه‌نویسی مشترکی وجود داشته باشد تا بتوان به راحتی در سامانه‌های غیرمتجانس مختلف برنامه‌نویسی کرد.

به هر حال GCC در برخی موارد کارآیی مورد نیاز روی یک بستر را ندارد. در نتیجه شرکتهای دیگر مترجمهایی ارائه کرده‌اند که برای یک بستر خاص بهینه شده است. اگر کارا بودن برنامه کاربردی برایتان اهمیت زیادی دارد، استفاده از مترجمهای تجاری اغلب نتایج بهتری دربردارد. ولی این، احتمالاً به قیمت از دست دادن قابلیت انتقال تمام شود. اگر درگیر توسعه هسته لینوکس هستید یا می‌خواهید بسترهای متعددی را تحت پوشش قرار دهید، GCC بهترین انتخاب است.

GCC توسط شرکت Cygnus – که مایکل تایمن (Michael Tiemann) مؤسس آن بود – تجاری شد. او یکی از اولین مدلهای تجاری متن باز را ارائه داده است. اگر چه GCC متن باز است، Cygnus آن را خرید و برای آن پشتیبانی و خدمات ارائه داد. Cygnus اکنون به RedHat پیوسته است.

Perl^۱

Perl که مخفف Practical Extraction Report Language به معنای "زبان استخراج گزارش عملی" است در حقیقت یک زبان برنامه‌نویسی تفسیری مختص پردازش متن است. Perl از سال ۱۹۸۷ مورد استفاده قرار گرفته است و طرفداران زیادی داشته و در نتیجه متخصصین زیادی هم دارد. از این زبان اغلب برای مدیریت سامانه، آپاچی و برنامه‌های کاربردی تحت وب استفاده می‌شود. از آنجایی که برنامه‌های کاربردی تحت وب اغلب با پردازش اطلاعات مبتنی بر متن سروکار دارند، این زبان رویکرد و راه حل مناسبی برای این گونه کاربردها است. اگر نگاهی به سازمان خود بیاندازید، احتمالاً تعدادی برنامه‌نویس آشنا با Perl پیدا خواهد کرد. چون Perl برای طیف وسیعی از سامانه‌های عامل قابل دسترس است رویکردهای مناسبی مستقل از بستر برای فعالیتهای توسعه‌ای فراهم می‌کند.

Python^۲

Python نیز یک زبان تفسیری شیء گرا می‌باشد که تحت بسترهای مختلفی موجود است. Python به اندازه Perl طرفدار ندارد، ولی زبانی است که توسعه‌دهندگان شیء گرا مثل کسانی که با C++ آشنا هستند به سرعت یاد می‌گیرند. این زبان نیز با پردازش متن سروکار دارد، ولی

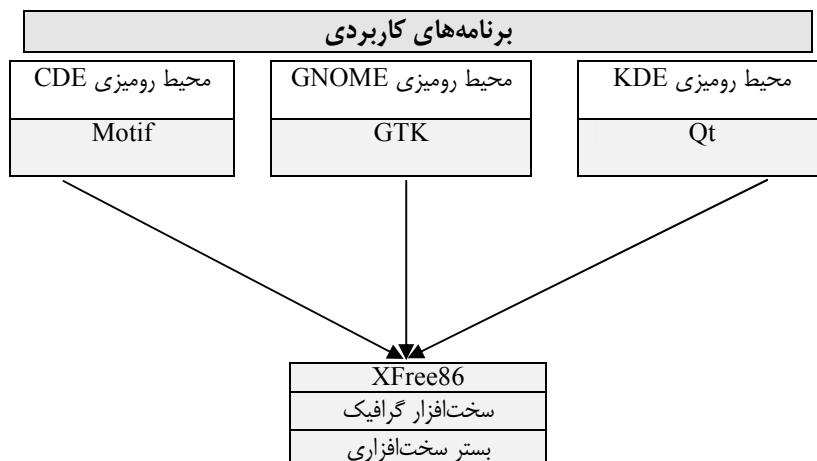
¹ <http://www.perl.org>

² <http://www.python.org>

در عین حال گزینه مناسبی برای نمونه‌سازی سریع کاربردهای مبتنی بر واسطه‌های گرافیکی است. از Python نیز مانند Perl اغلب در برنامه‌های کاربردی تحت وب استفاده می‌شود.

۴-۴- محیط‌های رومیزی و برنامه‌های اداری

یکی از بزرگترین تفاوت‌های بین لینوکس و ویندوز وجود چند میز کار گرافیکی در لینوکس و یک میز کار در ویندوز است که توسط مایکروسافت ارائه می‌شود. بعضی معتقدند که این تعدد محیط‌های رومیزی باعث سردرگمی می‌شود و هزینه‌های اضافی آموزش و پشتیبانی تحمیل می‌کند. بعضی هم بر این باورند که این پدیده ابتکار و ناآوری بوجود آورده است. هر دو میزهای کار لینوکس جوان هستند و زندگی خود را از سال ۱۹۹۷ آغاز کردند. شکل ۴-۴ نحوه تعامل محیط‌های رومیزی با کتابخانه گرافیکی XFree86 را نشان می‌دهد.



شکل ۴-۴: سلسله مراتب فناوری محیط‌های رومیزی

همچنین پروژه‌ایی به منظور ارائه نرم‌افزارهایی، مشابه مجموعه نرم‌افزارهای اداری مایکروسافت (MS Office) به شکل متن‌باز شکل گرفته‌اند که مشهورترین آنها OpenOffice می‌باشد.

XFree86^۱

XFree86، اصلی‌ترین سامانه نمایش پنجره‌ها است که در همه سامانه‌های لینوکس از آن استفاده می‌شود. این سامانه، پیاده‌سازی متن‌باز چارچوب کاری X11 است که توسعه آن از دانشگاه MIT آغاز شد. XFree86، امکان دسترسی سطح پایین به سخت‌افزار گرافیک نصب

^۱ <http://www.xfree.org>

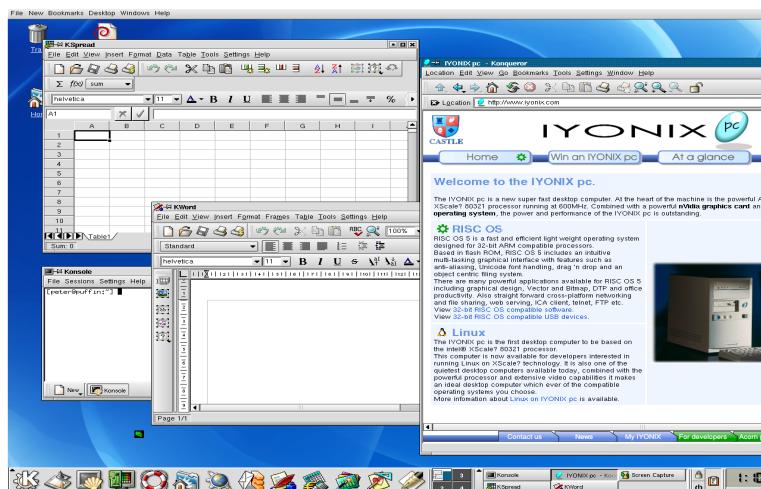
شده روی سامانه را فراهم می‌کند. اگر از ایستگاه‌های کاری یا میزکارهای لینوکس استفاده می‌کنید، سخت‌افزار گرافیک نصب شده روی سامانه شما باید از XFree86 پشتیبانی کند. اگر چه امکان نوشتن برنامه‌های کاربردی کاملاً کارآ برای X11 وجود دارد ولی پیچیدگیهایی به همراه دارد که باعث می‌شود این کار به ندرت انجام شود. در عوض از کتابخانه‌های نمایش پنجره سطوح بالاتر که با X11 در تعامل هستند، استفاده می‌شوند. کتابخانه‌ای که برای سالهای زیادی مورد استفاده قرار می‌گرفت Motif بود. محیط رومیزی که به همراه Motif استفاده می‌شد، CDE بود. در مورد لینوکس، توسعه‌دهندگان نیاز به کتابخانه‌های نمایش پنجره و محیط‌های رومیزی متن‌باز داشتند. برای نیل به این هدف دو پروژه عمده شکل گرفت: KDE و GNOME¹.

پروژه میزکار گنوم اواخر تابستان سال ۲۰۰۰ توسط بنیاد GNOME کلید خورد. گنوم توسط شرکت Novell [سابق Helox Code] Ximian تجارتی شد [شرکت مذکور توسط شرکت خردباری شد]. میزکار گنوم توسط میکوئل دو ایکازا برای ارائه میز کار کاملاً متن‌باز در سال ۱۹۹۷ آغاز شد. هر میزکار مبتنی بر مجموعه‌ای از کتابخانه‌های واسط گرافیک کاربری (GUI) است.

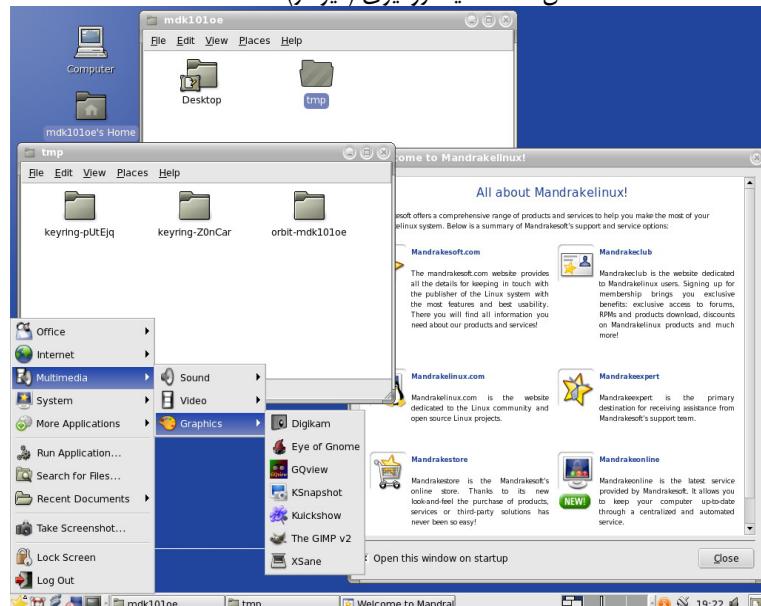
در آن زمان، KDE مبتنی بر کتابخانه‌ای بود که تماماً متن‌باز نبود. کتابخانه نمایش پنجره برای نوشتن برنامه‌های کاربردی گنوم، GTK و برای KDE، Qt نام دارد. شکل ۴-۵ نمایش صفحه‌ای از میزکار گنوم است.

مؤسسه گنوم با همکاری [NOVELL, Ximian, RedHat, IBM, HP, SUN توسعه میزکار گنوم را رهبری می‌کند. این پروژه بستر توسعه گنوم برای برنامه‌نویسان GNOME Office را نیز شامل می‌شود. بسیاری از عرضه‌کنندگان یونیکس نیز میزکار گنوم را به عنوان میزکار اصلی محیط‌های ایستگاه کاری یونیکس خود انتخاب کرده‌اند. اگر مایلید میزکار همگونی در سامانه‌های یونیکس و لینوکس خود داشته باشید، گنوم انتخاب خوبی خواهد بود.

¹ <http://www.gnome.org>



شكل ٥-٤ محیط رومیزی (میز کار) KDE



شكل ٦-٤ محیط رومیزی (میز کار) گنوم

KDE^۱

KDE نیز محیط رومیزی مشهوری برای لینوکس است. بحث درباره اینکه کدام میز کار بهتر است، اغلب سلیقه‌ای است. در واقع این بحث بیشتر برمی‌گردد به اینکه کاربر کدام را ترجیح می‌دهد یا چه کاری می‌خواهد انجام دهد یا یک مشخصه خاصی که در یکی موجود است و در دیگری وجود ندارد. شکل ۴-۶ نمایش صفحه‌ای از میز کار KDE است.

همان‌گونه که گنوم مؤسسه گنوم را به منظور گردهم آوردن رهبران صنعتی برای کمک و سرمایه‌گذاری در فرآیند توسعه گنوم، بنا نهاد، لیگ KDE نیز با اهدافی مشابه تشکیل شد.

موزیلا^۲ (Mozilla)

موزیلا یک مرورگر وب است که وارث مرورگر نت اسکیپ (Netscape) است. موزیلا در حقیقت اسم رمز اصلی مرورگر نت اسکیپ بود و وقتی شرکت نت اسکیپ تصمیم به متن‌باز کردن این محصول گرفت از اسم رمز آن استفاده کرد.

این گونه عرضه نت اسکیپ را می‌توان بازترین مثال از عرضه یک محصول تجاری تحت مجوز متن‌باز دانست. این کار در سهای خوبی به جامعه متن‌باز داد. چند سال زمان لازم بود تا موزیلا بتواند کارآیی و قابلیت اعتمادی را که ما در نت اسکیپ و مرورگر اینترنت مایکروسافت سراغ داشتیم، بدست آورد.

با اقدام مایکروسافت به عرضه^۳ IE به عنوان یک برنامه رایگان همراه ویندوز، عرضه موزیلا تسریع پیدا کرد. این رویدادها هم‌مان بود با انتشار مقاله معروف اریک ریموند با عنوان "کاتدرال و بازار" هم‌مان بود.

شرکت نت اسکیپ بر این باور بود که می‌تواند از روش توسعه اشتراکی که اریک در مقاله‌اش توصیف کرده است، برای پیشبرد توسعه یک مرورگر جایگزین استفاده کند. اما این ایده نت‌اسکیپ درست از آب درنیامد. آنها مرتکب چندین اشتباه شده بودند (مثل عرضه کدی که کاملاً کارآ نیست). قسمت سوم این کتاب تماماً در مورد پیشگیری از این خطاهای دیگر است.

امروزه موزیلا در زمرة مرورگرهای وب مطرح و قابل اعتماد قرار دارد.^۴

Open Office^۵

اگرچه چندین پروژه طراحی مجموعه نرم‌افزارهای اداری متن‌باز در حال توسعه است، پروژه OpenOffice قابل اطمینان‌ترین جایگزین برای مجموعه مایکروسافت آفیس شناخته شده است.

¹ <http://www.kde.org>

² <http://www.mozilla.org>

³ Internet Explorer

⁴. امروزه بنیاد موزیلا تبدیل به مهد چندین پروژه متن‌باز شده است، از جمله تاندربرد (Thunderbird) که یک نرم‌افزار ایمیل است. این بنیاد اولین نسخه جدای مرورگر موزیلا را نیز تحت نام فایرفاکس (firefox) در سال ۲۰۰۴ عرضه کرد.

⁵ <http://www.openoffice.org>

این پروژه پیاده‌سازی متن‌باز محصول StarOffice از شرکت SUN Microsystems است که امروزه موجود می‌باشد.

در سال ۲۰۰۰ میلادی شرکت SUN به منظور ارائه میزکار اداری برای سامانه‌های یونیکس و رقابت با مایکروسافت، شرکت StarDivision را خرید و بصورت زیرمجموعه خود درآورد. SUN محصول StarOffice را به صورت مخصوصی ثبت شده و پشتیبانی شده ارائه می‌کند در حالی که یک OpenOffice سامانه کاملاً آزاد است.

۴-۵- پایگاه داده‌ها

پایگاه داده‌ها زیربنای اصلی بسیاری از برنامه‌های کاربردی را تشکیل می‌دهد. این سامانه‌ها توابعی برای ذخیره‌سازی، سازمان‌دهی و بازیابی اطلاعات به شکل بهینه ارائه می‌دهند. وقتی نام پایگاه داده‌ها را می‌شنویم نامهای Oracle، Sybase و DB2 به یاد می‌آوریم. گرچه این پایگاه داده‌های تجاری، مقیاس‌پذیری فوق العاده، کارایی و ویژگیهای زیادی را فراهم می‌کنند، سامانه‌های متن‌بازی هم وجود دارند که این نیازها را به شکل بهتر و با قیمت کمتر، برآورده می‌کنند.

PostgreSQL^۱

یک موتور پایگاه داده حرفه‌ای و شی‌گرای مبتنی بر SQL است که سابقه آن به اواخر دهه ۱۹۷۰ و اوایل دهه ۱۹۸۰ در دانشگاه برکلی برمی‌گردد. اجداد قبلی آن با پیاده‌سازی‌های تجاری شرکت‌های Ingres (که بعدها به Computer Associates پیوست) و Informix (که به IBM پیوست) شکل گرفتند. با درنظر داشتن چنین سابقه‌ای، در اکثر کاربردهای پایگاه دادها می‌توان به طور جدی روی PostgreSQL حساب کرد. برخی شرکتها پشتیبانی‌های تجاری برای این پایگاه داده‌ها ارائه کرده‌اند.

MySQL^۲

پایگاه داده معتبر متن‌باز دیگری است که اغلب در برنامه‌های کاربردی تحت وب و کاربردهای شخصی از آن استفاده می‌شود. به هر حال MySQL پایگاه داده‌ای با کارایی بالا است که طرفداران زیادی دارد و خود را به عنوان پراستفاده‌ترین پایگاه داده متن‌باز معرفی می‌کند. افراد زیادی را به سمت خود جلب کرده است.

این پایگاه داده همچنین مثال عملی خوبی از مجوزدهی دوگانه است. همه می‌توانند پایگاه داده MySQL را با رعایت شرایط GPL به کار گیرند. به هر حال اگر شما نیاز به یک مجوز تجاری برای محصول خود دارید و قصد دارید پایگاه داده‌ای را همراه با محصولتان ارائه دهید، MySQL گزینه مناسبی است.

¹ <http://www.PostgreSQL.org>

² <http://www.MySQL.org>

۴-۶- PDA^۱

ارائه یک لیست کامل و جامع از همه پروژه‌های متن‌بازی که نرم‌افزارهایی برای بسترهای قابل انتقال طراحی و توسعه می‌دهند، ممکن نیست. اگر به پیرامون خود بنگرید می‌توانید برنامه‌های کاربردی متن‌باز متعددی برای PocketPC، PalmOS و تعداد دیگری از بسترهای لینوکس پیدا کنید. در اینجا تعدادی از پروژه‌های اصلی متن‌باز که روی لینوکس برای سامانه‌های دستی^۲ تمرکز کرده‌اند، آورده شده است.

۳- Handheld Linux

جامعه متن‌باز در حال توسعه سامانه‌های عامل و نرم‌افزارهایی برای بسترهای دستی است. بیشتر کاری که امروزه انجام می‌شود بر روی IPAQ متمرکز شده است زیرا IPAQ، سامانه‌عامل خود را که PocketPC OS نام دارد در حافظه فلش ذخیره می‌کند و این امر به توسعه‌دهندگان امکان جایگزینی PocketPC OS را با سامانه‌عامل خودشان می‌دهد. HP نیز به‌طور غیر رسمی از این گونه کارهای توسعه‌ای متن‌باز حمایت می‌کند. صنعت به سرعت در حال انتشار جایگزینهای PDA مبتنی بر این روش توسعه متن‌باز است. حتی تعدادی توزیع کاملاً توسعه‌یافته برای این بسترهای تهیه شده است.

۴- ۷- خوشه‌ها^۴

از خوشه‌ها برای تغییر مقیاس سامانه‌ها به صورت افقی، برای رسیدن به کارآیی بیشتر و فراهم کردن درجه بالاتری از دسترس‌پذیری برای کل سامانه استفاده می‌شود. شرکتهایی وجود دارند که خوشه‌های متن‌باز را پشتیبانی می‌کنند و پیاده‌سازی‌های مختلف خوشه‌ها را به شکل تجاری در می‌آورند.

خوشه‌ها Beowulf، برجسته‌ترین راه حل در زمینه خوشه‌های لینوکس، برای محاسبات سنگین، می‌باشد.

Beowulf^۵

پروژه Beowulf مثال فوق العاده‌ای از نحوه تأثیر اقتصاد محصولی^۶ بر دنیای فناوری اطلاعات است. هدف این پروژه فراهم کردن امکان محاسبات سنگین با استفاده از محصولات متن‌باز می‌باشد. Bewulf، راه حلی نرم‌افزاری برای خوشه‌بندی تعداد کثیری از گره‌ها^۷ جهت محاسبات موازی ارائه می‌دهد.

¹ Personal Digital Assistants

² Handheld

³ <http://www.handhelds.org>

⁴ Clusters

⁵ <http://www.beowulf.org>

⁶ Commodity

⁷ Node

اگر در گیر پروژه‌هایی هستید که می‌توانند از معماری‌های محاسبه موازی بهره‌گیرنده، Beowulf نقطهٔ فوق‌العاده‌ای برای آغاز کار است.

۴- سازمانها

رشد و شهرت لینوکس و جنبش متن‌باز مدیون سازمانهای زیادی است. اغلب این سازمانها، موسسات غیرانتفاعی هستند که قصدشان بدست آوردن سود مالی نیست و وقت و انرژی خود را به ارتقاء جنبه‌های مختلف توسعه و گسترش متن‌باز اختصاص داده‌اند. تعداد کثیری از این سازمانها با سرمایه و بودجه اندکی کار می‌کنند و نیازمند کمک و حمایت هستند. کمکها می‌توانند به صورت اهداء تجهیزات، منابع، خدمات بازاریابی و مبالغ نقدی باشد.

^۱ Linux International

Linux International، سازمانی غیرانتفاعی است که لینوکس را در سطح بین‌المللی ترویج می‌دهد و توسط جان هال² (John Maddog Hall) مدیریت می‌شود. این سازمان، لینوکس را به رهبران تجاری می‌شناساند و مزایای لینوکس را برای استفاده‌های تجاری عنوان می‌کند. همچنین منابع مالی پروژه‌هایی را که نزد جامعه لینوکس ارزش و اهمیت بیشتری دارند، تأمین می‌کند.

^۲ بنیاد نرم‌افزار آزاد

همان‌طور که قبلاً اشاره شد، هدف FSF، ترویج و ارتقاء نرم‌افزار آزاد برای عموم است. FSF زایدۀ ذهن ریچارد استالمن و منزلگاه پروژه گنو و مجوز GNU GPL است. اگر می‌خواهید نسخه‌ای از متن GPL را در اختیار داشته باشید یا از جزئیات نرم‌افزار آزاد کسب اطلاع کنید، باید به اینجا مراجعه کنید.

^۳ Open Source Initiative

OSI در حقیقت OSD را ترویج و ارتقاء داده و فهرستی رسمی از مجوزهای مورد قبول و مصوب متن‌باز را نگهداری می‌کند. در زمان نوشتن این کتاب، چیزی بیش از ۳۰ مجوز تصویب شده وجود دارد. با توجه به اینکه طراحی مجوز جدید امکان پذیر است، قبل از اینکه مجوز متن‌باز جدیدی طراحی کنید بهتر است نگاه دقیقی به مجوزهای موجود بیاندازید.

^۴ آزمایشگاه توسعه متن‌باز

وب‌گاه^۵ OSDL، توسط جمعی از رهبران صنعتی با انگیزه ارتقاء لینوکس برای مراکز داده‌ها و ارتباطات راه دور تشکیل شد. OSDL، آزمایشگاه پیشرفته‌ای در اختیار دارد که به هرگونه منابع و

¹ <http://www.li.org>

² <http://www.fsf.org>

³ <http://www.opensource.org>

⁴ Open Source Development Lab

⁵ <http://www.osdl.org>

ابزاری - که ممکن است برای آزمون سنگین یک پروژه متن باز نیاز باشد - مجهز است. هر فردی می‌تواند برای فرستادن پروژه خود به آزمایشگاه و دسترسی به منابع آن درخواست کند. OSDL برآورده از کارآیی درخواستی افراد به عمل می‌آورد و همواره تلاش می‌کند بر این اساس تجهیزات خود را تکمیل کند. اگر در زمینه تسریع و شتابدهی به آماده شدن قابلیتهای لینوکس برای مراکز داده و محیطهای ارتباطات راه دور انگیزه دارد، OSDL انجمن خوبی برای آشنا شدن است.

گروه استانداردهای آزاد^۱

فصل هفتم جزئیات کاملی از استانداردهای ارائه شده توسط این گروه را شرح می‌دهد.^۲

در حال حاضر گروههای کاری اصلی زیر را شامل می‌شود:

- پایه استانداردهای لینوکس (LSB)

LSB، مجموعه‌ای از استانداردهای واسط برای اطمینان از قابل انتقال بودن روی بسترها متن باز را تعریف می‌کند.

- بین‌المللی سازی ya Li18nux

Li18nux در زمینه ایجاد استانداردهایی برای ترویج استفاده لینوکس در بازارهای بین‌المللی تلاش می‌کند.

کنسرسیوم لینوکس توکار^۳

لينوکس به سرعت در کاربردهای توکار محبوبیت پیدا کرد و امروزه تعدادی از شرکتها و توزیعهای لینوکس فعالیت خود را روی لینوکس توکار مرکز کرده‌اند.

کنسرسیوم لینوکس توکار سازمانی تجاری است که به ارتقاء لینوکس برای کاربردهای توکار اختصاص دارد. اگر پروژه‌ای توکار را شروع کرده‌اید و لینوکس نیازهای کاری شما را برآورده می‌کند یا در جستجوی شرکتهای دیگر برای کار کردن با آنها هستید، از این سازمان یازدید کنید.

پروژه مستند سازی لینوکس^۴

واژه‌ای که اغلب در این انجمن به کار می‌رود، "HowTo" است. همان‌طور که از نام آن برمی‌آید، HowTo راهنمایی است که توسط افراد مختلف جامعه، در زمینه استفاده از لینوکس، نوشته شده است و در مورد هر موضوعی که فکر کنید مطلبی تهیه شده است. پروژه مستندسازی لینوکس تلاشی برای رسمی کردن مستندات لینوکس است.

همه می‌توانند در این پروژه شریک باشند. این پروژه در زمینه ترجمه راهنمایی موجود نیز فعالیت می‌کند.

¹ Free Standards Group (FSG)

² <http://www.freestandards.org>

³ <http://www.embedded-linux.org>

⁴ <http://www.linuxdoc.org>

۹-۴ جمع بندی

وقتی شما مطالعه این فصل را به پایان می‌رسانید باید کاملاً متقادع شده باشید که متن باز یک رویکرد گذرا نیست. آن قدر تعداد شرکتها، گروهها و افرادی که در ارتباط با آن هستند زیاد است که امکان کنار گذاشتن آن نیست. حرکتی که باید انجام شود، تضمین آینده‌ای روش برای طیف وسیع برنامه‌های کاربردی متن باز موجود است و هرچه تعداد آنها بیشتر شود سهم همتای تجاری آنها نیز از بازار کمتر می‌شود.

اگر شما این رده از برنامه‌ها را در محیط فناوری اطلاعات خود به کار می‌برید، باید نگاه جدی به این موضوع داشته باشید. اگر در هر یک از این زمینه‌ها توسعه‌دهنده هستید یا رقبای جدیدی دارید که انگیزه‌های اقتصادی یکسانی با شما ندارند، اینجاست که شما باید به جای رقابت با آنها از مزایای آنها استفاده کنید و مدل تجاری خود را به سطح جدیدی برسانید. بخش سوم کتاب چگونگی این کار را به شما خواهد آموخت.

این فصل را می‌توان آخرین بخش آموزش مقدماتی نیز تلقی کرد. اکنون شما اطلاعات کافی برای چگونگی گسترش لینوکس و متن باز در سازمان خود دارید. بخش دوم کتاب تماماً در مورد به کارگیری آنچه که تا اینجا یاد گرفتیم، در کسب و کار است.

بخش دوم لینوکس در عمل

با اطلاعاتی که تاکنون در مورد لینوکس به دست آورده‌اید، می‌توانید این مبحث را آغاز کنید که پیاده‌سازی و گسترش لینوکس در سازمان و تجارت در عمل به چه معنی است. لینوکس یک پروژه متن‌باز خیلی بزرگ است. اگر چه این بخش روی گسترش جنبه‌های لینوکس تاکید دارد، با این حال قسمت عمده آن قابل اعمال به دیگر نرم‌افزارهای متن‌باز نیز هست.

بخش دوم با نگاه مفصلی به توزیعهای لینوکس آغاز می‌شود. توزیعها برای ارائه یک سامانه‌عامل مفید و کارآ، روی هسته لینوکس ساخته می‌شوند. بعد، نگاهی به تاثیرات استفاده از لینوکس و متن‌باز در تجارت خواهیم انداخت و همان طور که پیش می‌رویم استانداردهای مرتبط با لینوکس را مرور می‌کنیم. بخش دوم با بررسی اجزاء عملی به کارگیری، انتقال از یک بستر یا سامانه‌عامل به بستر یا سامانه‌عامل دیگر، هم زیستی^۱، پشتیبانی و آموزش به پایان می‌رسد.

^۱ Coexistense

فصل پنجم: توزیعها-تکمیل لینوکس

اگر در حال تهیه خودرویی هستید که با آن از نقطه A به نقطه B بروید، هیچ وقت تنها موتور خودرو را نمی خرید. دریافت و ساخت هسته لینوکس را می توان مشابه در اختیار داشتن موتور دانست. بدیهی است که شما نیاز به یک خودروی کامل دارید و بسته به نیازтан ممکن است خودرو سواری، وانت و یا هر وسیله‌ای از بی شمار وسیله نقلیه موجود را انتخاب کنید. حتی وقتی به خوبی می دانید که چه می خواهید، ممکن است امکانات اضافه‌ای را فراتر از نیاز خود در خودرو سفارش دهید.

توزیعهای لینوکس مشابه آن اتوموبیل هستند. توزیعها، هسته لینوکس را با قابلیتها و ویژگیهای دیگر ترکیب می کنند. برخی از توزیعها مختص کاربردهای گسترده موجود در بازار هستند و برخی دیگر بر نیازهای مکانهای جغرافیایی و یا محلهای خاص تمرکز می کنند. یکی از پیش‌نیازهای اصلی برای درک لینوکس، فهم جزئیات و ساختار یک توزیع است. در این فصل خواهید آموخت:

- یک توزیع چیست؟
- بسته‌ها و وابستگیها
- عرضه‌کنندگان اصلی توزیعها
- چه چیز یک توزیع را از توزیع دیگر متفاوت می کند.
- مواردی که باید هنگام انتخاب یک توزیع لحاظ شوند.
- وابستگی به یک توزیع
- نگاهی گذرا به استانداردهای مرتبط با توزیعها

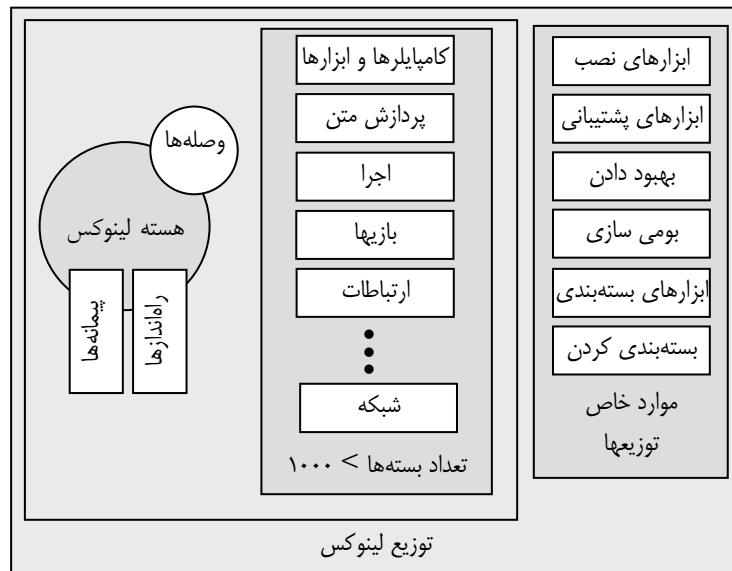
آنچه که شما در شرکت خود به کار می گیرید و گسترش می دهید، توزیعهای لینوکس هستند. اگر برنامه‌های کاربردی تولید می کنید، درک تفاوت‌های میان توزیعها تضمین خواهد کرد که شما بتوانید تمامی محیط‌های ممکن را پشتیبانی کنید.

۱-۵- توزیع لینوکس

اکثر مردم وقتی به "لینوکس" اشاره می کنند، عموماً منظورشان توزیعهای لینوکس است. همان طور که به زودی خواهید آموخت، یک توزیع، توسط یک عرضه‌کننده تهیه می شود. آنها مؤلفه‌ها و اجزای متعددی را با هسته همراه می کنند. عرضه‌کنندگان توزیع عموماً وظایف زیر را انجام می دهند:

- یک نسخه خاص هسته را انتخاب می کنند.
- وصله‌هایی (Patch) را به منظور افزودن کارایی یا حل مشکلات طراحی می نویسند یا انتخاب می کنند.
- در زمینه استفاده از مجموعه خاصی از بسته‌های نرم‌افزاری و نسخه‌های آنها تصمیم‌گیری می کنند.

- کارهای ارزش افزوده (کارهایی که باعث افزایش ارزش توزیع می‌شود) خود را به آن می‌افزایند مثل نرم‌افزار نصب بهینه و توسعه‌ابزارها.
 - کل مجموعه را در یک رسانه (مثلاً لوح فشرده) جهت پخش ترکیب می‌کنند.
 - هر جزء و همچنین کل توزیع را آزمون و تایید نموده و از کارآیی آن اطمینان حاصل می‌کنند.
 - در برخی موارد از برنامه‌های کاربردی شرکتهای ثالث استفاده می‌کنند.
 - از نشانها و علامت تجاری خاص خود در توزیع استفاده می‌کنند.
 - در مورد خطمشیهای معرفی به بازار تصمیم‌گیری می‌کنند.
 - خدمات ارزش افزوده خود را همراه با توزیع ارائه می‌دهند.
- هر توزیع لینوکس با دیگری فرق دارد. شکل ۱-۵ دیدی از ساختاری که تقریباً در همه توزیعها مشترک است، نشان می‌دهد. همان طور که می‌بینید، یک توزیع خیلی بیشتر از فقط یک هسته است.



شکل ۱-۵: شمای کلی یک توزیع لینوکس

عرضه‌کنندگان توزیع کارهای بسیار زیادی برای تهیه یک لینوکس انجام می‌دهند. اینکه آنها این کار را چقدر خوب انجام می‌دهند و چقدر خوب می‌توانند برای کارهایشان پشتیبانی ارائه دهن، فاکتورهای اصلی در انتخاب یک توزیع، برای نصب می‌باشد.

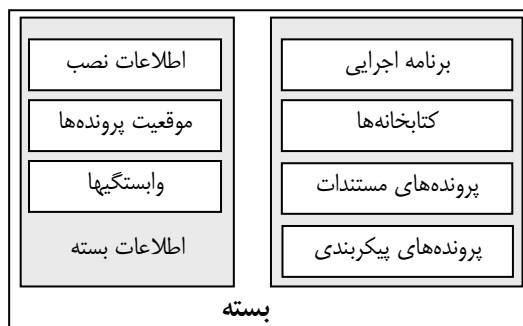
غلب عرضه‌کنندگان توزیع، مجموعه‌ای از بسته‌های نرم‌افزاری متن باز را با هم ترکیب می‌کنند. این بدین معنی است که توزیعهای آنها را می‌توان به صورت رایگان دریافت و نسخه‌برداری کرد. بنابراین عرضه‌کنندگان توزیعها مجبور شدن مدلهای تجاری خود را به منظور حفظ پیشرفت شرکت، تکامل بخشنده. اینجا خلاصه کوتاهی از نحوه کسب درآمد عرضه‌کنندگان توزیع آمده است:

- فروش مجموعه رسانه و مستندات در یک جعبه از طریق کانالهای خرد فروشان
- ارائه خدمات پشتیبانی و به روزرسانی
- ارائه خدمات حرفه‌ای مرتبط با توزیع لینوکس خود
- دریافت هزینه از تولیدکنندگان تجهیزات که توزیع را با محصولات نرم‌افزاری یا سخت‌افزاری خود ارائه می‌کنند.

عرضه‌کنندگان توزیع به طور پیوسته مدلهای تجاری خود را بهبود می‌بخشند و بر فهرست زیر بدون شک افزوده خواهد شد. در بخش سوم کتاب به جزئیات مدلهای تجاری لینوکس و متن باز خواهیم پرداخت.

۲-۵- بسته‌های نرم‌افزاری

هر توزیع نماد "بسته نرم‌افزاری" را به همراه دارد، زیرا یک توزیع همه بسته‌های نرم‌افزاری تعیین شده برای آن نسخه را گردآوری می‌کند. هر بسته از یک یا چند پرونده تشکیل شده که در حالت کلی یک وظیفه (مثل یک مترجم، یک مرورگر، یک کارساز وب یا غیره) را انجام می‌دهد.



شکل ۲-۵: ساختار عمومی یک بسته

برای روشن تر شدن موضوع باید اشاره کرد که این نگهدارنده بسته است که درمورد چیدمان بسته تصمیم می‌گیرد. برخی بسته‌ها بسیار بزرگ هستند و شامل هزاران پرونده را می‌شوند در حالی که برخی دیگر خیلی کوچکند و از بیش از یک یا دو پرونده تشکیل نمی‌شوند. شکل ۲-۵ طرح کلی یک بسته را نشان می‌دهد.

همان طور که می‌بینید، یک بسته هم پروندهایی مورد نیاز برای به انجام رساندن وظایف خود و هم اطلاعاتی را که مورد نیاز مدیر بسته برای نصب بسته روی سامانه مقصد است را در بر می‌گیرد.

فهرست جدول ۱-۵ از بسته پایدار دییان گرفته شده است. دیگر عرضه کنندگان توزیع ممکن است این فهرست را به اشکال دیگری گروه‌بندی کنند. بدینهی است که فهرست کردن همه بسته‌ها ممکن نیست (فقط دییان از حدود ۹۰۰۰ نرمافزار تشکیل شده است) ولی جدول ۱-۵ دسته‌بندی اصلی بسته‌ها را فهرست کرده است.

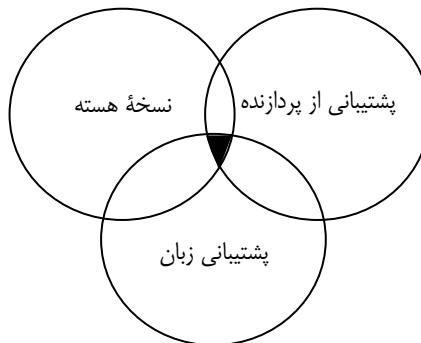
جدول ۱-۵ دسته بندی نرمافزارهای بسته‌ها

نام دسته	شرح
برنامه‌های مدیریتی	برنامه‌های کمکی برای راهبری منابع سامانه، مدیریت کاربران و انجام سایر امور راهبری سامانه
برنامه‌های ارتباطی	نرمافزارهای ارتباطی از طریق یک مودم سنتی
توسعه و برنامه‌نویسی	ابزار و محیط‌های برنامه‌نویسی، کامپایلرها و کتابخانه‌ها
مستندات	سوالهای متداول (FAQ) راهنمایها و غیره
ویرایشگرها	نرمافزارهای ویرایش پروندها
الکترونیکی	نرمافزارهای الکترونیکی
بازی	سرگرمی
گرافیک	ویرایشگرها، نمایش دهندها و مبدلها
رادیویی آماتوری	نرمافزارهای کار با رادیوهای آماتوری
تفسیرها	هر نوع مفسری برای زبانهای تفسیرشونده و پردازشگرهای ماکروها
کتابخانه‌ها	کتابخانه‌ها برای کار برنامه‌ها مورد نیاز هستند و امکانات خوبی در اختیار برنامه‌نویسان قرار می‌دهند.
نامه	برنامه‌های مسیردهی، خواندن و نوشتن پیامهای پست الکترونیک
ریاضیات	نرمافزارهای ریاضی
متفرقه	برنامه‌های کمکی متفرقه که در دسته خاصی نمی‌گنجند

نام دسته	شرح
شبکه	دیمونها (برنامه‌های کارخواه) که در پس زمینه اجرا می‌شوند) و کارسازها (مانند telnet و مرورگر وب) که لینوکس را به دنیای خارج متصل می‌کنند.
گروههای خبری News	نرم‌افزارهای دسترسی به یوزنوت و کارسازهای اختراعی (News) اکثر بسته‌های نرم‌افزاری را می‌توان در ایالات متحده استفاده کرد ولی قابل صادر کردن نیستند (حتی نمی‌توان مجدداً آنها را وارد کشور کرد). برخی از این نرم‌افزارها ممکن است به خاطر مسائل مربوط به حق ثبت، در ایالات متحده و یا کشورهای دیگر قابل استفاده نباشند. باید با دقت محدودیتهای نرم‌افزارها را مطالعه و در صورت لزوم با وکیل خود مشورت کنید.
کتابخانه‌های قدیمی	نسخه‌های قدیمی کتابخانه‌ها که برای سازگاری با نسخه‌های قبلی برنامه‌ها حفظ می‌شوند.
سامانه‌های عامل و پرونده	نرم‌افزارهای اجرا کننده برنامه‌های سایر سامانه‌های عامل و استفاده از سامانه پرونده آنها سامانه‌های دیگر
پوسته‌ها	پوسته‌های فرمان رابطه‌ای کاربرپسند برای افراد مبتدی
صوتی	نرم‌افزارهای کمکی مدیریت صدا: میکسرهای، اجرا و پخش کننده‌ها، ضبط صوتها
TeX	نرم‌افزار معروف حروف‌چینی و برنامه‌های مربوط با آن
پردازش متن	برنامه‌های کمکی قالب بندی و چاپ اسناد متنی
برنامه‌های کمکی	برنامه‌های کمکی برای کار با دیسک/پروندها، ابزار بایگانی و ساختن نسخه پشتیبان، نظارت بر عملکرد سامانه و سامانه‌های ورودی
نرم‌افزار وب	کارسازهای وب، مرورگرهای، پراکسی‌ها و ابزار دریافت پرونده
نرم‌افزار سامانه	کارسازهای X، کتابخانه‌ها، فونتهای، مدیریت پنجره‌ها، برنامه‌های ترمینال و برنامه‌های دیگر در این زمینه X Window

هر بسته‌ای نگهدارنده یا گروهی از نگهدارندهای دارد و از نسخه‌های مختلفی تشکیل می‌شود. اکثر بسته‌ها بومی‌سازی شده‌اند و می‌توان آنها را به هر زبانی یافت. عرضه کنندگان تولید نه تنها در مورد اینکه چه بسته‌هایی باید با توزیعشان ارائه شود، بلکه در مورد نسخه‌های بسته‌ها، وصله‌های همراه با این بسته‌ها و زبانهایی که باید پشتیبانی شوند، نیز تصمیم می‌گیرند.

یک عرضه‌کننده توزیع برای اینکه بسته‌ای را با یک توزیع همراه کند دقت می‌کند که آن بسته، نسخه هسته مورد استفاده را پشتیبانی کند، نسخه بسته برای پردازنده‌ای که پشتیبانی می‌شود موجود باشد و زبان انتخابی هم پشتیبانی گردد. این گزینه‌ها، اصلی‌ترین موارد برای در نظر گرفتن و لحاظ شدن هستند. اگر چه ممکن است موارد دیگری هم در برخی شرایط در نظر گرفته شود. نمودار ون در شکل ۳-۵ شرایط مورد نیاز برای اینکه یک بسته در توزیع قرار گیرد را نشان می‌دهد.



شکل ۳-۵: دیاگرام ون مشترکات بسته‌ها

همیشه باید به خاطر داشته باشید که یکی از ویژگیهای کلیدی متن باز این است که شما به عنوان مشتری کنترل را در دست دارید نه عرضه‌کننده. اگر از تصمیماتی که فراهم‌کننده توزیع‌تان گرفته است راضی نیستید، می‌توانید هر اندازه تعییر که مایل هستید روی آن اعمال کنید. اگر بسته خاصی مورد نظرتان است که در توزیع ارائه نشده است، می‌توانید آن را در اینترنت پیدا کنید و یک نسخه از آن را برای نصب دریافت کنید.

۱-۲-۵ - قالب^۱ بسته

بسته‌ها از ترکیب پرونده‌ها ساخته می‌شوند و با به کارگیری ابزارهای ویژه‌ای که پیمانه‌های خاصی تولید می‌کنند، آماده استفاده می‌شوند. دو قالب مشهور بسته وجود دارد: یکی توسط Red Hat Package Manager یا RPM و توسعه‌یافته است که نام برد. اگر بخواهید فقط نامیده می‌شود و دیگری متعلق به انجمن دیبان است و از آن به فرمت DEB (یا دات دب) یاد می‌شود.

۳-۵ - عرضه‌کنندگان توزیعها

تعداد عرضه‌کنندگان لینوکس چنان زیاد است که نمی‌توان همه آنها را نام برد. اگر بخواهید فقط توزیعهای خاص یا مناسب نیازهای محلی را بشمارید، تعداد آنها به صدها عدد می‌رسد. در اینجا

¹ Format

روی توزیعهایی که فرآگیرتر هستند (سهم عمده‌ای از بازار را در اختیار دارند) تاکید می‌شود و مثالهایی از توزیعهایی که بر بازارها یا مکانهای جغرافیایی خاص و محدود مرکز دارند، نیز آورده خواهد شد. می‌توانید فهرست کاملی از توزیعها را در وبگاه [linux.org](http://distrowatch.org) مشاهده کنید.

۱-۳-۵ - توزیعهای فرآگیر

RedHat

در گوشه و کنار جهان ردهت شناخته شده‌ترین توزیع لینوکس است و یکی از چند شرکت موجود تجاری است که کاملاً روی لینوکس مرکز دارد. باب یانگ (Bob Young) و مارک ایوینگ (Marc Ewing)، ردهت را در سال ۱۹۹۴ تشکیل دادند. ردهت یکی از اولین نمونه‌هایی بود که نصب و استفاده از لینوکس را آسان کرده بود. ردهت، سهم عمده‌ای در بازار دارد ولی در همه مناطق جهان نفوذ نکرده است.

ردهت از توزیع خود به عنوان یک کارت اعتباری معتبر استفاده می‌کند که در بردارنده سابقه آن در خدمت به لینوکس است. آنها تعدادی از توسعه‌دهندگان را -که به طور مستقیم با لینوکس و دیگر اجزاء سامانه‌عامل سروکار دارند- به کار می‌گیرند و در زمینه‌های بسیاری از قبیل RPM ابداعاتی کرده‌اند.

وقتی مدل‌های تجاری را با جزئیات بیشتری بررسی می‌کنیم، خواهید دید که ردهت، اکثر انرژی خود را صرف توسعه خدمات پشتیبانی تجاری (غیررایگان) کرده است. آنها از همان ابتدای کار دریافتند که با فروش مجوز نرم‌افزارهای آزاد نمی‌توانند درآمد مناسبی برای ادامه کار شرکت ایجاد کنند. با این وجود، ردهت نسبت به متن‌باز و فادر مانده است و از همراه کردن و عرضه هرگونه نرم‌افزاری که منطبق با تعریف متن‌باز نیست، امتناع ورزیده است. معلوم نیست تاچه زمانی می‌توانند با این روش ادامه دهند و تنها زمان، مشخص کننده این مسئله است.

همچنین باید یادآوری کرد که ردهت، Cygnus Solution را در اختیار خود درآورد که نقش کلیدی نگهدارنده ابزارهای مترجم GCC را بر عهده دارد.

ترکیب یکی از مشهورترین توزیعهای لینوکس با مجموعه‌ای از خدمات و پشتیبانی، ردهت را به نقش آفرینی با اعتبار در دنیای لینوکس تبدیل کرده است.

SUSE

این توزیع که در آلمان پایه‌ریزی شده است، توزیعی است که می‌تواند روی طیف وسیعی از معماری‌های ریزپردازنده استفاده شود. SUSE بواسطه پایه آلمانی‌اش، در منطقه اروپا از محبوبیت خاصی برخوردار است [شرکت SuSE در ۱۳ ژانویه ۲۰۰۴ توسط شرکت آمریکایی ناول خریداری شد]. این توزیع، اغلب رهبر فناوری لینوکس خوانده می‌شود و می‌تواند هم در محیط‌های کارساز و هم در محیط‌های کارخواه به کار گرفته شود و به عنوان بستر اصلی توسعه بسیاری از تولید-کنندگان نرم‌افزار استفاده شود. ردهت، اصلی‌ترین رقیب آن محسوب می‌شود. برخی جنبه‌های مدل تجاری آن خیلی شبیه به ردهت است در حالیکه در برخی جنبه‌ها کاملاً متفاوت است. مرکز SUSE بر پشتیبانی چندین بستر و دسته‌بندی و فروش مجدد نرم‌افزارهای شرکتهای ثالث

به همراه لینوکس باعث شده است که مدل تجاری آن به گونه دیگری رشد کند. این مسأله برای آنها جریان درآمدی از جانب تولیدکنندگان تجهیزات اصلی (OEM) و مجوزهای نرمافزار ایجاد کرده است.

Caldera

Caldera، روند تکامل جالبی در دنیای لینوکس داشته است. این شرکت که خود را وقف لینوکس کرده بود، به SCO (از تولیدکنندگان یونیکس) ملحق شد و توانست به پشتونه قوی تری برای فعالیت خود تکیه کند. اکنون Caldera، هم محصولات لینوکس و هم محصولات یونیکس تولید می کند و روی مسائل اساسی یکپارچه سازی بین این دو رویکرد تمرکز کرده است. caldera در حال توسعه محصولات ارزش افزوده برای ارائه به همراه توزیع خود است تا بتواند علاوه بر خدمات حرفه ای و پشتیبانی، از مجوز توزیع خود هم درآمد کسب کند.

Turbo Linux

این توزیع بیشترین نمود را در بازارهای آسیایی داشته است و اگر چه RedHat در این زمینه سایه به سایه Turbo Linux پیش می رود و تهدیدی برای آن به شمار می رود، با این حال هنوز هم Turbo Linux در این بازار قدرت اول است. یکی از ویژگیهایی که Turbo Linux با توزیع خودش ارائه داده است توسعه زبان آسیایی است که همین مسأله باعث محبوبیت آن شده است. Turbo Linux، تاکنون چندین مدل تجاری مختلف را امتحان نموده که از جمله آنها می توان به تلاش ناکام این شرکت در ادغام با Linux Care اشاره کرد.

اکنون Turbo Linux یک مدل تجاری دارد که بر ارائه راه حل هایی در زمینه قابلیت مدیریت، خوشبندی و پایگاه داده ها تمرکز دارد. به علاوه این شرکت توانسته است توزیع خود را برای چندین ریزپردازنده مختلف آماده کند.

Mandrake

Mandrake حیات خود را با شعار "یک رد هت بهتر" شروع کرد. این شرکت کار خود را با یک نسخه از توزیع رد هت آغاز کرد و ویژگیهای مفیدی را به آن افزود و یک توزیع -که برای آخرین نسل پردازنده های اینتل ترجمه و بهینه شده بود- تهیه کرد. Mandrake، با گذشت زمان از رد هت فاصله گرفت. امروزه این توزیع به عنوان یک لینوکس رومیزی قوی مطرح است و تلاش می کند جای خود را در میان سامانه های کار ساز به عنوان رویکردی قابل اطمینان باز کند. Mandrake در فرانسه پایه ریزی شده که این امر آن را در میان البته بعد از رد هت اروپائیان محبوب ساخته است [این شرکت در سال ۲۰۰۵ با شرکت بزری Convectiva ادغام شد و نام جدید Mandriva را برگزید].

Debian

دیان بواسطه نداشتن ارتباطات تجاری، توزیعی قابل توجه است. این توزیع، تنها توزیعی است که ۱۰۰٪ مبتنی بر انجمنها است. ایان مرداک (Ian Murdock) مؤسس دیان است. نام دیان تلفیق نام همسرش Debby و نام خودش Ian است.

این خصیصه که دییان مبتنی بر ارتباطات تجاری نیست، دییان را میان رهبران و توسعه - دهندگان جامعه متن باز محبوب ساخته است. انجمن دییان شامل صدها توسعه دهنده است که جهت عمدۀ فعالیت آنها در راستای دیگر توزیعهای فراگیر است. همچنین به یاد آورید که پیشینه OSD نیز از قرارداد اجتماعی دییان (debian.org/social_contract) می باشد.

دییان به این می بالد که نسبت به سایر توزیعهای بیشترین بستر را پشتیبانی می کند. شرکتهای زیادی مایل به همکاری با دییان هستند، زیرا در اینجا فرآیند توسعه هیچ زمانبندی ندارد و کارهای عرضه شده به محض حاضر شدن پذیرفته می شوند و از آنجایی که همه چیز باز و آزاد است در محیطی برابر رقابت می کنند. اگر عرضه کننده نرم افزار هستید و به استانداردسازی مؤلفه های خاص زمینه کاری خود نیاز دارید، دییان می تواند بهترین محیط برای کار کردن باشد.

بسیاری از انجمنها و سازمانها دییان را به عنوان پایه ای برای کارشان به کار می گیرند، سپس برای دیگر توزیعهای خدمات پشتیبانی فراهم می کنند. دییان به واسطه کم کاری در پشتیبانی جدیدترین نوآوریها در بسته ها و طراحی های هسته مورد انتقاد قرار گرفته است. دییان همواره به خاطر عقب بودن از سایر توزیعهای در پشتیبانی از ابداعات جدید در نرم افزارها و نسخه های جدید هسته مورد انتقاد واقع شده است. ولی این در واقع شمشیری دو لبه است. گرچه ممکن است دییان تا حدی عقب تر از دیگر توزیعهای حرکت کند، اما به عنوان توزیعی شناخته شده است که کامل ترین آزمایشها روی آن صورت گرفته است و روی گسترده ترین طیف از بسترها و با بیشترین پایداری کار می کند.

شرکتهایی که دریی بازترین توزیع با حداقل وابستگی به تهیه کننده و حداکثر پایداری هستند دییان را انتخاب می کنند. تنها نکته منفی در مورد دییان این است که به واسطه فقدان پشتیبانی شرکتی تولید کنندگان نرم افزارهای تجاری این توزیع را آخرین انتخاب قرار می دهند.

United Linux

United Linux (UL) همکاری تعداد زیادی از عرضه کنندگان پیشرو توزیعهای لینوکس برای ایجاد یک مبنای استاندارد برای لینوکس یا ایجاد یک لینوکس منطبق بر LSB است (در فصل ۷ در این مورد بحث خواهد شد). از آنجایی که ساخت یک توزیع کار پرهزینه ای است و برای عرضه کنندگان، ارائه محصولات برای توزیعهای متفاوت، مشکل می باشد، تعداد زیادی از عرضه کنندگان برای ساخت یک توزیع واحد لینوکس گردهم آمدند. آنها می توانند با همسوسازی تلاشهای خود هزینه های ساخت توزیع پایه را بین هم تقسیم کنند و انرژی خود را بر کارهای بالرزش دیگری متمرکز کنند.

در زمان نوشتن این کتاب UL کارهای ابتدایی خود را شروع کرده است و برای پیش بینی درباره این که آیا UL در صنعت به عنوان یک توزیع رایج محبوب خواهد شد یا خیر خیلی زود است. خوب بخانه این تلاشها باعث تعیت از LSB و سازگاری برنامه های کاربردی با این استاندارد خواهد شد.

-۲-۳-۵ - توزیعهای محلی

برخی توزیعها نیازهای خاص بازارهای منطقه‌ای را پاسخ می‌دهند. به چندین مورد از آنها به عنوان مثالهایی از این دست اشاره خواهد شد. ولی نمی‌توان فهرست کاملی از این توزیعها ارائه نمود. با توجه به جائی که زندگی می‌کنید، ممکن است توزیعی محلی وجود داشته باشد که نیازهای شما را براورده سازد.

Red Flag

این توزیع می‌کوشد که سامانه‌عامل لینوکس را به کشور بزرگ چین معرفی کند. Red Flag هم لینوکس توکار و هم لینوکس فرآگیر را تهیه کرده است. پیشرفت این لینوکس در راستای نیازهای بازارهای آسیای جنوب شرقی می‌باشد.

اگر تولیدکننده برنامه‌های کاربردی لینوکس هستید و تصمیم دارید چین را بازار هدف محصولات خود قرار دهید، لازم است با یک فراهم کننده محلی لینوکس مثل Red Flag ارتباط برقرار کنید.

Connectiva

این توزیع به عنوان یک توزیع مختص آمریکای لاتین شناخته شده است. Connectiva در برزیل پایه‌ریزی شده و به زبانهای اسپانیایی و پرتغالی توزیع می‌شود. Connectiva نیز همانند اغلب تولیدکنندگان لینوکس، یک سری خدمات پشتیبانی و آموزشی ارائه می‌کند که بر کسب وکارهای موجود در بازار آمریکای لاتین تمرکز دارد. [این شرکت در سال ۲۰۰۵ با شرکت Mandrake ادغام شد.]

-۳-۳-۵ - توزیعهای اختصاصی

مثالهای زیر نحوه تمرکز تولیدکنندگان توزیع بر بازارهای خاص را نشان می‌دهند. اکثر این تولیدکنندگان در یک زمینه خاص تبحر دارند.

MSC.Linux

شرکت MSC.Software، در زمینه برنامه‌های کاربردی محاسباتی با کارآیی بالا فعالیت می‌کند و در زمینه کاری‌اش شناخته شده است. توزیع MSC، مختص کاربردهای شبیه‌سازی، خوشبندی و کارآیی بالا است. MSC به همراه توزیع خود، خدمات ساخت‌افزاری، مشاوره و ساختن توزیعهای سفارشی را نیز ارائه می‌کند و در نهایت سعی می‌کند یک سامانه کاری بهینه برای محیط‌های کاربردی خاص مشتریان تهیه کند.

MontaVista

MontaVista (که نام اولیه توزیع آنها Hard Hat Linux بود) بر بازارهای لینوکس توکار و ارتباطات راه دور تمرکز دارد و توزیعهایی تهیه می‌کند که قابلیت اجرای برنامه‌های کاربردی روی پردازنده‌های کوچک و توکار را دارد. همچنین با افزودن ویژگیهای پیشرفته به هسته، امکان ارائه قابلیتهای بلاذرنگ را فراهم می‌کند. MontaVista، چندین نوع پردازنده توکار را پشتیبانی می‌کند، محیط توسعه‌ای (برنامه‌نویسی) مبتنی بر کاربردهای توکار فراهم می‌کند.

همچنین این شرکت به پشتیبانی از چندین نوع برد مادر رایانه تخصصی (مثل PCI و VME) مبادرت کرده است.

۴- توزیعهای سامانه‌های عامل غیرلینوکس

واژه "توزیع" برای اشاره به هرگونه بسته‌بندی از پروندها، برنامه‌ها، ابزارها و خدمات به کار می‌رود که لزوماً شامل سامانه‌عامل لینوکس نیست. بهترین مثال از این مورد توزیع میزکار گنوم است. گرچه یک توزیع لینوکس اغلب از مجموعه‌ای از بسته‌های نرمافزاری تشکیل شده است که میزکار گنوم را نیز شامل می‌شود، اما ممکن است عرضه کنندگان تجاری بسته نرمافزاری بهبود یافته و ثبت شده‌ای تحت عنوان "توزیع گنوم"، طراحی و ارائه کنند. در ادامه مثال میزکار گنوم، Ximian نیز توزیعی طراحی کرده است که موارد زیر را با هم ترکیب می‌کند:

- بسته‌های مورد نیاز برای اجرای گنوم
- ابزارهای پیشرفته و مخصوص توسعه‌دهندگان
- کد منبع
- نرمافزارهای اداری
- قابلیتهای بهروزرسانی نرمافزار
- نسخه‌هایی برای سامانه‌های عامل مختلف (مثل UX-HP و سولاریس)
- رسانه و مستندات

البته این فقط یک مثال است. تعدادی از شرکتهای بازرگانی تلاش در طراحی و ساخت توزیعها برای هر تعداد از اجزای سامانه‌عامل لینوکس دارند. بعضی تلاشها موفق‌تر از بقیه هستند. نکته کلیدی که باید به خاطر داشته باشید این است که گرچه واژه "توزیع" اغلب در مورد لینوکس به کار می‌رود، اما ممکن است به دیگر مجموعه‌های نرمافزارهای کاربردی نیز اطلاق شود.

۵- طراحی توزیع برای خودتان

بسیاری از شرکتها به این نتیجه می‌رسند که نیاز به طراحی و ساخت توزیع دلخواه خود برای برآوردن نیازهای خاص سازمان خود دارند. به هر حال فرآیند طراحی یک توزیع نباید ساده فرض شود. بسته به گستردگی تغییرات و ویژگیهای مورد نظر ممکن است متتحمل هزینه زیادی شوید. نوعاً توزیعهای طراحی شده متناسب با نیازها و خواسته‌ها با موارد زیر بنا می‌شوند:

- یک هسته لینوکس خاص همراه با وصله‌ها و پیمانه‌های از پیش تعیین شده
- اجزای مربوط به نامهای تجاری ثبت شرکتی
- ابزارها و قابلیتهای خاص شرکت شما
- عاملهای مدیریتی از پیش تنظیم شده
- برنامه‌های کاربردی از پیش نصب شده که دارای مجوز هستند
- شبکه پیکربندی شده و مؤلفه‌های دیگر
- تمهیدات امنیتی منطبق با استانداردهای شرکت

- قابلیتهای خاص برای تعامل با قسمتهای دیگر سازمان
 - پشتیبانی سخت افزارهای خاصی که در شرکت استفاده می شوند
 - حذف همه بسته های غیر ضروری
 - یک هسته و تعدادی بسته نرم افزاری که ترجمه شده و برای اجرا روی سخت افزاری خاص بهینه شده اند.
 - چند برنامه نصب کننده توزیع حاصل، برای محیط های مختلفی که در سازمان مورد استفاده هستند (میز کار، کارخواه ها، کارخواه های چند پردازنده و غیره)
- طرایی توزیع خودتان زمانی باید انجام شود که سازمان فناوری اطلاعات شما آن قدر توسعه یافته باشد که آمادگی مدیریت تغییرات مداوم همه بسته های تشکیل دهنده توزیع را داشته باشد. به مقوله خود اتکایی باید همانند یکی از پروژه های نرم افزاری سازمان نگریسته شود. شکل



شکل ۵-۳: ساختن یک توزیع لینوکس

۴-۵ دو روند کار متفاوت برای ساخت یک توزیع دلخواه را نشان می دهد. فرآیند سمت چپ با یکی از توزیعهای فرآگیر آغاز شده، مؤلفه های نامطلوب را حذف می کند سپس مؤلفه های جدید را اضافه می کند. فرآیند سمت راست با یک هسته لینوکس آغاز و توزیع مورد نظر روی آن ساخته می شود. اینکه کدام فرآیند را انتخاب کنید بستگی به نیازها و درجه کنترل و تغییرات مورد نظر شما دارد.

۵-۶- پشتیبانی از چندین توزیع

یکی از انتخابهای کلیدی برای شرکتهای بزرگ چند ملیتی تصمیم در مورد این مسأله است که آیا شرکت یک توزیع را برای همه کاربردها پشتیبانی خواهد کرد یا چندین توزیع برای کاربردهای مختلف ارائه خواهد داد.

شما به عنوان یک عرضه‌کننده نرمافزار، همواره خواهان هرچه بزرگتر کردن بازار محصولات خود هستید. وقتی تصمیم به گسترش برنامه‌های کاربردی خود روی لینوکس می‌گیرید، ممکن است تلاش کنید توزیعی را انتخاب کنید که با آن بیشترین آشنایی را دارد. در این بین ممکن است یکی از مشتریان از شما بخواهد که از توزیع رایج دیگری یا حتی توزیعی که به دلخواه تغییر یافته است، پشتیبانی کنید. استانداردها به این منظور گسترش یافته‌اند که اختلافات بین توزیعها، تأثیر منفی روی عرضه‌کنندگان نرمافزار نداشته باشد. شما باید برای تعییت از آن استانداردها حداقل تلاش را بکنید. از آنجایی که استانداردها، نقش کلیدی در زمینه طراحی توزیعها و نرمافزارها دارند، در فصل هفتم به طور مفصل به آنها خواهیم پرداخت.

روش دیگری که عرضه‌کنندگان نرمافزار استفاده می‌کنند، این است که به جای تمرکز بر وابستگی به توزیعها، به وابستگی بر مؤلفه‌های یک توزیع تأکید دارند. لیستی از وابستگیها مانند زیر تهیه کنید:

- نسخه هسته
- وصله‌های مورد نیاز
- بسته‌های مورد نیاز
- نسخه‌های تمامی بسته‌ها
- وابستگیها ببرونی برنامه‌ها
- نیازمندیهای سخت‌افزاری

این لیست جامع، باعث حذف وابستگیها به یک توزیع پایه می‌شود و امکان پشتیبانی هر توزیعی را که بازار تقاضا کند به شما می‌دهد. این مسأله نیاز شما به آزمایش برنامه‌های کاربردی خود تحت ترکیبات مختلف توزیعها و سخت‌افزارها را مرتفع نمی‌کند، ولی شما را مطمئن می‌سازد که خود را وابسته یک توزیع خاص که فرصت‌های بازاری آینده شما را محدود خواهد کرد، نساخته‌اید.

۷-۵- استانداردها

فصل هفتم نگاه مفصلی به استانداردهای مرتبط با لینوکس و توزیعها خواهد داشت. عموماً همه توزیعها مبتنی بر یک هسته مشترک و مجموعه مشترکی از بسته‌ها هستند، ولی به هر حال هر توزیع ممکن است انتخابهای متفاوتی در زمینه نسخه‌های این مؤلفه‌ها و آنچه که باید در کار شامل شود و یا جزء کار نباشد داشته باشد. اگر تصمیم به اجرای لینوکس روی معماریهای ریزپردازنده غیرفراغی، دارید، ممکن است بسته‌های کمتری نسبت به لینوکسهای نصب شده در سامانه‌های رایج در اختیار داشته باشید.

تعدادی استاندارد در حال توسعه است که تضمین می‌کند توزیعها بتوانند بر ارزشها و کاربردهای خاص خود بیفزایند و در عین حال عرضه کنندگان سخت‌افزاری و برنامه‌های کاربردی بتوانند به سادگی و به‌طور یکسان همه توزیعها را پشتیبانی کنند.

۸-۵- جمع بندی

اکنون شما باید درک روشی از گزینه‌های پیش رو در زمینه توزیعی که می‌خواهید در محیط کاری خود استفاده کنید، داشته باشید. توزیعها مجموعه‌ای از بسته‌های نرم‌افزاری هستند که در کنار هم قرار می‌گیرند و برای اطمینان از اینکه این مجموعه از نسخه‌ها همگی به خوبی با هم کار می‌کنند، آزمایش می‌شوند. توزیعها امکان نصب و استفاده مجموعه‌ای از نرم‌افزارهای مناسب کسب و کار شما را فراهم می‌کنند.

در فصل بعدی نگاهی به هزینه‌های لینوکس خواهیم انداخت. گرچه می‌توان از توزیعها بدون پرداخت هزینه‌ای استفاده کرد، اما استفاده و پشتیبانی مؤثر آن، در طول چرخه حیات سامانه برای شما هزینه‌هایی دربرخواهد داشت. نحوه محاسبه این هزینه‌ها به طور قابل توجهی با نحوه محاسبه شما در گذشته فرق دارد.

فصل ششم:

هزینه لینوکس و متن باز

لینوکس رایگان است، ولی نه ۱۰۰ درصد. هر مدیر با تجربه‌ای می‌داند که هزینه‌های مؤلفه‌ها تنها بخش کوچکی از کل هزینه‌ها هستند و هزینه کل رویکرد باید با درنظر گرفتن چرخه حیات آن، محاسبه شود. واژه معادل صنعتی که در این زمینه استفاده می‌شود «هزینه کل مالکیت»^۱ یا TCO است. ماشینی را که در فصل گذشته می‌خواست شما را از نقطه A به نقطه B ببرد، به خاطر می‌آورید؟ بسته به نوع ماشین، نیاز دارید که هزینه سوت، بیمه، نگهداری، اجزاء و عوامل دیگر را در طول چرخه عمر این وسیله لحاظ کنید. این امر در مورد تهیه فناوریهایی مثل متن باز نیز صادق است، مخصوصاً اینکه در مورد متن باز، آزاد بودن آن به هزینه نداشتن تعبیر می‌شود.

در این فصل مؤلفه‌های تأثیرگذار در هزینه‌های متن باز را بررسی خواهیم کرد و ممکن است متفاوت از هر محاسبه دیگری (محاسبات مربوط به هزینه) باشد که شما در گذشته داشته‌اید. به خاطر داشته باشید که لینوکس یک پروژه متن باز خیلی عظیم است. گرچه ممکن است تمایل داشته باشید در زمینه هزینه‌های مربوط به لینوکس تمرکز کنید، اما مواردی که توضیح داده می‌شوند، مجازاً قابل استفاده در هر پروژه متن باز دیگر نیز هستند.

اهداف این فصل عبارتند از :

- فهم و درک هزینه کل یک رویکرد
- تأثیرات متن باز روی هزینه‌های رویکرد
- نحوه بدست آوردن نرم‌افزارهای متن باز
- چه فرآیندهای تجاری نیاز دارند که از نوآزموده شوند

این فصل اطلاعاتی را در اختیار شما می‌گذارد که هنگام محاسبه هزینه کل مربوط به استفاده و اجرای یک محیط لینوکس مورد نیاز است.

۶-۱- هزینه‌ها

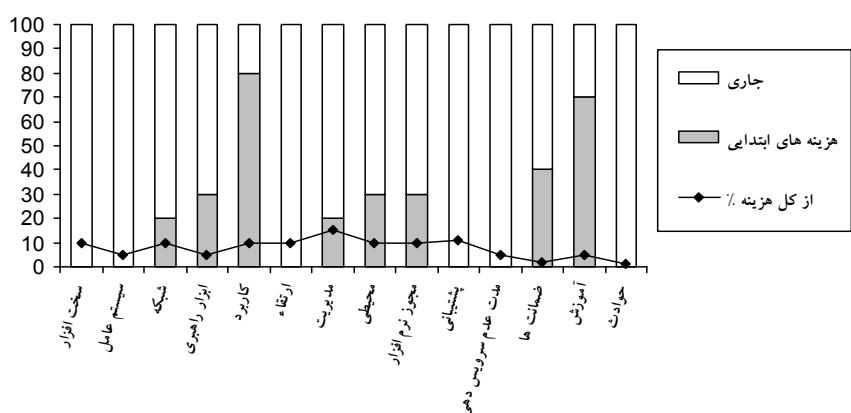
لینوکس و دیگر نرم‌افزارهای متن باز نحوه اندازه‌گیری هزینه یک رویکرد را متحول کرده‌اند. در زیر برخی از هزینه‌های سنتی مربوط به رویکردهای مبتنی بر فناوری اطلاعات آورده شده است:

- سخت‌افزار
- سامانه‌عامل
- زیرساخت شبکه
- ابزارهای مدیریت و کمکی
- استفاده
- ارتقاء رویکرد در حال استفاده

^۱ Total Cast Ownership

- مدیریت و راهبری پیوسته شامل ردیابی مجوزها
 - شرایط محیطی (فضای کاری، برق مورد نیاز، تهویه و غیره)
 - مجوزهای برنامه‌های کاربردی
 - هزینه‌های پشتیبانی (برای همه مؤلفه‌ها)
 - خرابیهای پیش‌بینی نشده
 - تعهدات و تضمینهای تمدید شده
 - آموزش
 - حوادث ناگوار و دیگر ریسکهای تجاری

وقتی که محاسبات مربوط به میزان «بازگشت سرمایه»^(ROI) را انجام می‌دهید، باید هزینه‌های اولیه و هزینه‌های نگهداری پیوسته را نیز به طور دقیق مشخص کنید. مثلاً هزینه خرید مجوز اولیه یک برنامه کاربردی جدید در قیاس با هزینه‌های پشتیبانی -که این برنامه کاربردی در طول چرخه حیاتش دارد- می‌تواند ناچیز باشد. از طرف دیگر عرضه کنندگان اغلب برخی از این مولفه‌های هزینه را (مثل سختافزار با سامانه‌عامل) با هم عرضه می‌کنند که این کار محاسبه و مقایسه را مشکل می‌سازد. شکل ۱-۶ یک نمونه از هزینه‌های نسبی به کارگیری رویکردهای رایانه‌ای در یک سازمان فرضی را به طور نظری نشان می‌دهد.



شکل ۶-۱: نمودار هزینه نمونه در پیاده‌سازی یک راه حل ریاضی‌ای

¹ Return of Investment

چون هر سازمان با سازمان دیگر متفاوت است، شکل ۱-۶ تنها تلاش می‌کند که یک سناریوی محاسبه هزینه نوعی را ارائه دهد (در سازمان شما ممکن است اساساً متفاوت باشد). این نمودار، درصدهای نسبی هزینه‌های گسترش یک رویکرد کامل را نشان می‌دهد. نمودار میله‌ای درصد هزینه اولیه تهیه یک مؤلفه از رویکرد را نسبت به هزینه‌های مربوط به نگهداری آن مؤلفه در حین پیشرفت کار، نشان می‌دهد. به عنوان مثال، هزینه‌های شرایط محیطی شامل برخی هزینه‌های اولیه (ساخت یک اتاق رایانه، نصب تجهیزات برق و تهویه و غیره) می‌شود. ولی در عین حال هزینه‌های محیطی دیگری مربوط به نگهداری مؤلفه‌های نیز موجود می‌باشد که در حین پیشرفت کار شکل می‌گیرد. قسمت خاکستری رنگ میله هزینه‌های ثابت اولیه را نشان می‌دهد. قسمت سفیدرنگ میله هزینه‌های در حال پیشرفت را نشان می‌کند. نمودار خطی درصد نسبی این دو هزینه را برای هر مؤلفه از کل رویکرد نشان می‌دهد.

همان گونه که ملاحظه می‌کنید اکثر مؤلفه‌ها هم هزینه‌های ثابت اولیه و هم هزینه‌های متغیر در جریان کار دارند. در اکثر موارد هزینه‌های در جریان کار، درصد بیشتری از هزینه‌های ثابت را به خود اختصاص می‌دهند. همچنین می‌توان مشاهده کرد که هیچ کدام از مؤلفه‌ها قسمت بزرگی از کل هزینه رویکرد را به خود اختصاص نمی‌دهند. پیچیدگی که در اینجا نشان داده نشده است این است که اکثر سازمانها چندین رویکرد را به کار می‌گیرند و بسیاری از هزینه‌های مؤلفه‌ها با توجه به این مسأله با هم جمع خواهند شد. تحلیل تجاری شما باید این پیچیدگی را به شمار آورد. مهمترین فاکتور این است که اگر در حال استفاده از چندین رویکرد هستید و می‌توانید هزینه‌های ثابت را یک جا کنید، در آن صورت مؤلفه‌های متأثر از متن‌باز می‌توانند تاثیر جدی روی صرفه‌جوئی در هزینه‌ها داشته باشند.

اثر متن‌باز

از آنجایی که شناختن دقیق هزینه‌ها یکی از مهمترین گامها در به کارگیری لینوکس و نرم‌افزارهای متن‌باز است، شایسته است که به بررسی هر یک از این عناصر فهرست هزینه‌ها که متن‌باز بر آنها تأثیر مثبت یا منفی دارد، پپردازیم. در فصول بعدی برخی از این عناصر با جزئیات بیشتری بررسی می‌شود.

سخت‌افزار

نرم‌افزارهای متن‌باز به روش‌های گوناگونی بر هزینه‌های سخت‌افزار تاثیر می‌گذارند. در اینجا خصوصیات سامانه‌عامل لینوکس نقش کلیدی بازی می‌کنند. همان طور که در فصلهای ۲ و ۱ اشاره شد، به طور نظری همه نرم‌افزارهای متن‌باز شامل لینوکس روی تجهیزات رایانه‌ای کالایی باز^۱ طراحی شده‌اند و چون در این حالت احتیاجی به وابستگی به رویکردهای سخت‌افزاری اخلاقی نیست، هزینه سخت‌افزار می‌تواند به طور چشمگیری کاهش یابد. در این حالت فرض می‌شود که ترکیب یک بستر (سخت‌افزاری) باز و لینوکس نیاز به مقیاس‌پذیری، قابلیت اطمینان،

¹ Open Commodity Computing

در دسترس بودن و کارایی را برآورده می‌کند. نکته قابل توجه دیگر، اکوسیستم سخت‌افزاری (تجهیزات سخت‌افزاری که با یکدیگر سازگارند) موجود در نتیجه استفاده از بستر باز است. این مسئله بدین معنی است که رده گسترهای از افزودنیهای سخت‌افزاری پشتیبانی شده (مؤلفه‌هایی که برای بالا بردن کارآیی به رایانه اضافه می‌شوند)، موجود می‌باشد. این روش، گزینه‌های موجود پیش روی شما را افزایش خواهد داد و تأثیرات رقابتی نیز هزینه‌های شما را کاهش خواهد داد.

بسیاری از کاربران اظهار داشته‌اند که ترکیب لینوکس و نرم‌افزارهای متن‌باز روی پیکربندیهای سخت‌افزاری خیلی قدیمی نیز به خوبی اجرا می‌شود. لینوکس اغلب به عنوان راهی برای استفاده مجدد روی سخت‌افزارهای قدیمی استفاده می‌شود. همچنین سخت‌افزارهای دارای کارآیی پایین نیز اغلب می‌توانند نیازهای کسب و کار شما را با هزینه‌های خیلی کمتری برآورده کنند. با این همه ممکن است سخت‌افزارهای اختصاصی دست دوم یا بخشی از سامانه موجود برای بهره‌مند شدن از مزایای لینوکس به کار گرفته می‌شود.
سامانه‌عامل^۱

هزینه استفاده از سامانه‌های عامل اختصاصی روی سامانه‌های بزرگ می‌تواند به راحتی به صدها هزار دلار برسد. اما لینوکس را می‌توان بدون هیچ هزینه‌ای (مربوط به مجوز) دریافت، نسخه برداری و نصب کرد. مسلماً مهمترین هزینه به نگهداری، ارتقاء و پشتیبانی سامانه‌عامل در طول چرخه حیات آن مربوط می‌شود که همراه با پیشرفت کار انجام می‌پذیرد. این مسأله را بعداً با تفصیل بیشتری بررسی خواهیم کرد. بدیهی است که انتظار می‌رود ترکیب سخت‌افزار و سامانه‌عامل نیازهای تجاری شما را برآورده کند.

همان گونه که در فصل ۵ توضیح داده شد، شما قادرید مؤلفه‌های مختلف را برای ساخت توزیع لینوکس دلخواه خود ترکیب کنید. به جای تهیه جداگانه مؤلفه‌های مورد نیاز، می‌توانید اجزای متن‌باز دلخواه را با سامانه‌عامل خود همراه کنید.

بسته به نوع کاربرد و اندازه سازمان، تغییر توزیعهای تجاری لینوکس درجهت نیازهای خاص سازمان، می‌تواند هزینه‌های جدیدی به شکل قراردادهای نگهداری و پشتیبانی در پی داشته باشد.

ابزارهای مدیریت و کمکی

تقریباً هر شرکت بزرگی مجموعه گسترهای از ابزارهایی خاص خود ایجاد می‌کند. در اکثر موارد این ابزارها سالها پیش توسعه یافته‌اند و توسعه‌دهنده مربوطه شرکت را ترک کرده است. در برخی موارد کد منبع این ابزارها و سرویسهای داخلی را نیز نمی‌توان پیدا کرد. اینجاست که

¹ Operating System

هزینه‌های نگهداری و انتقال این ابزارهای داخلی به بسترهای دیگر می‌تواند بسیار قابل توجه باشد.

زیبایی متن باز اینجا نمایان می‌شود. تقریباً هر عملکردی که فکرش را بکنید توسعه-دهنده‌ای در جامعه متن باز نوشته شده است. اگر هم دقیقاً آنچه می‌خواهید موجود نباشد، چیزی نزدیک به آن یافت می‌شود که می‌توانید آن را متناسب با نیازهای خود تغییر دهید. قسمت جالب این ماجرا این است که وقتی شما ابزاری را توسعه می‌دهید، می‌توانید آن را به جامعه برگردانید تا آن را به طور پیوسته نگهداری کند. بدین ترتیب فرصت عالی برای کاهش هزینه‌ها در اختیار شما قرار می‌گیرد. همکاری با جامعه متن باز حسن نیت شما را نشان می‌دهد و در مقابل جامعه را به یاری شما علاقه‌مند می‌نماید.

رویکرد ارتقاء

ارتقاء آنچه که از آن استفاده می‌کنید، همیشه با شرایط و زمان‌بندی دلخواه شما انجام می‌شود. هیچگاه لازم نیست نگران از دست دادن پشتیبانی یک عرضه‌کننده باشید. چون کد منبع همیشه موجود است، هر وقت که بخواهید می‌توانید خودتان اقدام به پشتیبانی محیط موجود در سازمان کنید. وقتی که با پروژه‌های متن باز بزرگی چون لینوکس و کارساز آپاچی سروکار دارید، می‌توانید از پشتیبانی شرکتهای متعددی که در این زمینه وجود دارد، بهره‌گیرید یا می‌توانید با حمایت جامعه متن باز خودتان این پشتیبانی را برعهده گیرید.

فناکتور دیگر در هزینه‌های ارتقاء، به مجوزهای ارتقاء مربوط می‌شود. بسیاری از عرضه‌کنندگان نرم‌افزار برای انجام به روزرسانی‌های کوچک هزینه‌های نگهداری را دریافت می‌کنند و در مورد ارتقاء به نسخه‌های جدید، هزینه‌های سنگینی مطالبه می‌کنند.

مدیریت و راهبری و رديابي مجوزها

وقتی که در فصل هشتم مقوله به کارگیری را با تفصیل بیشتری مورد بررسی قرار دهیم، خواهید فهمید که یکی از مواردی که در به کارگیری یک محیط لینوکسی باید لحاظ شود، توانایی جای دادن آن در زیرساخت موجود است. شرکتها سالانه میلیونها دلار فقط صرف رديابي مجوزهای نرم‌افزار خود می‌کنند. بسیاری از آنها ابزارهای پیچیده‌ای برای رديابي مجوزهای خود در سطح جهانی ایجاد می‌کنند، در حالی که شرکتهایی هم وجود دارند که برای توافق با فروشنده‌گان در زمینه مجوزهای شرکتی و به حداقل رساندن هزینه‌های رديابي مجوز حاضر به انجام هر کاری هستند. در مورد نرم‌افزارهای متن بازی که تحت مجوزهای قابل قبول برای OSI توزیع شده‌اند، نیازی به رديابي مجوزها نیست و اگر هر تعداد نسخه از نرم‌افزار را که نیاز دارید نسخه برداری کنید کاملاً قانونی است.

مجوزهای برنامه‌های کاربردی و دیگر نرم‌افزارها

واضح است که یکی از بهترین جاهایی که می‌توانید هزینه‌ها را کاهش دهید، هزینه مجوز نرم‌افزار است. برای بسیاری از افراد این یک مفهوم جدید است. در اکثر موارد شما واقعاً نرم‌افزار متن باز را خریداری می‌کنید. نمونه آن خرید یک توزیع لینوکس بصورت بسته‌بندی شده از یک

فروشگاه است. ولی تفاوت عمدۀ اینجاست که شما هزینه‌ای بابت مجوز آن نمی‌دهید؛ در حقیقت شما بول راحتی بدست آوردن نرم‌افزار (به‌جای دریافت آن از اینترنت) می‌دهید. زیرا در این صورت این نرم‌افزار در شکل‌های مختلف رسانه (DVD، CD برای بارکردن نرم‌افزار) به همراه مستندات و اختیار شما قرار می‌گیرد.

وقتی مقوله پشتیبانی را با تفصیل بیشتری بررسی کنیم با گزینه‌های موجود آشنا خواهید شد. در اینجا کافی است آگاه باشید که نیازی به پرداخت هزینه مجوز برای نرم‌افزارهای متن‌باز نیست. شما می‌توانید نرم‌افزار را از هر جایی دریافت کنید، لوح‌های فشرده دلخواه خود را بسازید و حتی از مستندات را روی کاغذ چاپ کنید.

هزینه‌های پشتیبانی

یکی از متدالوں ترین مدل‌های تجاری مرتبط با متن‌باز، ارائه پشتیبانی نرم‌افزارهای متن‌باز است. وقتی با نرم‌افزارهای اختصاصی مواجه می‌شوید، گزینه‌های پشتیبانی به طور قابل توجهی محدود می‌شود و تنها می‌توانید از عرضه‌کنندۀ که نرم‌افزار را فروخته است، پشتیبانی بگیرید. در این حالت عرضه‌کننده، مشتری را سرگردانه گیرآورده و برای خدمات پشتیبانی خود تقاضای اجرت سنگینی می‌کند.

حتی اگر بنا دارید با نرم‌افزارهای متن‌باز به طریقی مشابه با نرم‌افزارهای تجاری برخورد کنید، این واقعیت ساده که خدمات پشتیبانی از طرف چندین عرضه‌کننده موجود است، رقابت را افزایش و تاثیر مثبتی روی کاهش قیمت دارد. علاوه‌شما نیاز به تغییر فرآیندهای تجاری خود برای در دست داشتن پشتیبانی به عنوان یک گزینه رقابتی دارید.

خرابیهای پیش‌بینی نشده

در دنیای نرم‌افزارهای اختصاصی، برخی نقاط ضعف می‌تواند سبب خرابیهای نامطلوب و غیر-منتظره شود. در مورد محیط‌های دارای ماموریتها و وظایف بحرانی، اغلب سرمایه‌گذاریهای هنگفتی برای به حداقل رساندن این موارد می‌شود. به هر حال شما در کنترل عرضه‌کننده هستید. اگر عرضه‌کننده تمایلی به اعمال تغییرات مورد نظر شما در بازه زمانی دلخواه شما نداشته باشد، معمولاً کاری از دستتان برنمی‌آید.

مجدداً حتی در شرایطی که شما با نرم‌افزارهای متن‌باز همانند نرم‌افزارهای تجاری کار می‌کنید، امکان استفاده آزادانه از جامعه متن‌باز یا در اختیار داشتن کادری از کارشناسان فناوری اطلاعات داخل سازمانی برای حل مشکلات، می‌تواند هزینه خرابیهای پیش‌بینی نشده را کاهش دهد.

آموزش

همان‌طور که در فصل ۱ اشاره شد، وجود منابع انسانی آموزش دیده، یکی از مزایای کلیدی حرکت رو به جلوی لینوکس است. اگر شما یک استاندارد صنعتی یا یک محیط متن‌باز را گسترش می‌دهید، هزینه‌های آموزش با توجه به این واقعیت که منابع انسانی آموزش دیده

فراوانی وجود دارند، کاهش می‌یابد. با این حال اگر در حال مهاجرت از محیط موجود هستید، هزینه‌های آموزش نیاز به سرمایه‌گذاری عظیمی دارند که باید لحاظ شوند. صرف نظر از نوع استفاده، نرم‌افزارهای متن‌باز، رفتارهای متفاوتی از جانب توسعه‌دهندگان، مذاکره کنندگان قراردادها، قادر خرید و تیم مدیریت می‌طلبند. این کتاب جزئیات مورد نیاز برای درک تفاوت این رفتارها را شرح خواهد داد، ولی خودتان باید هزینه‌های مربوطه را لحاظ کنید.

۶-۲- پذیرش یک رویکرد ناقص

در روزهای آغازین عصر رایانه، اکثر محیطها (راه‌حلها) از پایه ساخته می‌شدند. در نتیجه نیازهای سازمان را بخوبی برآورده می‌کردند. اما توسعه این راه‌حلها کار بسیار هزینه‌بری بود. به مرور زمان عرضه کنندگان نرم‌افزار، برنامه‌هایی کاربردی ساختند که مسائل و نیازهای زیادی را پوشش می‌دادند. اگرچه این راه حل‌ها کامل نبودند، اما هزینه تهیه آنها به طور چشمگیری کمتر بود، به طوری که انتخاب یک راه حل ناقص ترجیح داده می‌شد. به هر حال این هزینه‌های است که باید در کسب و کار خود لحاظ کنید و یکی از مهمترین مزایای متن‌باز این است که این هزینه‌ها را در بر ندارد.

اینجا نقطه درخشش متن‌باز است. توانایی در اختیار گرفتن نرم‌افزار متن‌بازی که بتواند نیازهای تجاری شما را تقریب بزند و سپس بتوانید آن را با نیازهای خاص خود تطبیق دهنده، بهترین گزینه ممکن است. همان‌طور که قبلاً اشاره شد در حالات زیادی جامعه متن‌باز از پیشرفتهای شما استقبال خواهد کرد؛ آنها را خواهد پذیرفت و کل محیط را پشتیبانی خواهد نمود.

به حساب آوردن این مزایا می‌تواند مشکل باشد، چرا که شما باید زمینه‌ای را تحلیل و مقایسه کنید که هزینه آن صفر است علاوه بر اینکه هزینه‌های افزاینده را نیز باید لحاظ کنید. با این تحلیل می‌توانید یک مورد تجاری بسازید. جدول ۱-۶ راهنمایی‌هایی برای این تحلیل ارائه می‌دهد:

جدول ۶- پیاده سازی یک راه حل "کامل"

هزینه های کاهش یافته	هزینه های جدید یا افزایش یافته	هزینه های حذف شده
پذیرفتن راه حل نرم افزار متن باز	به کارگیری راه حل های متن باز برای نیازهای تجاری	هزینه مجوز نرم افزارها در راه حل "ناکامل"
پذیرفتن کارهایی که نگهداری و پشتیبانی داخلی از هر جزئی که جامعه متن باز بروی نرم افزار انجام داده است	پرسنل اضافی در راه حل ناکامل	آن را نپذیرد
بروز رسانی پیوسته حفظ ارتباط با جامعه متن باز	راه حل های موقتی پرسنل برای تأثیرگذاری و توسط جامعه متن باز	سفارشی کردن
شروع کردن از کدهای منتشر شده موجود	پرسنل حرفه ای	تأثیرات تجاری دیگر (مانند زمان عرضه در بازار، رضایت مشتری و غیره)
کنترل کیفیت گسترده توسعه جامعه متن باز		

بديهی است که اين جدول مقاييسات هزينه، بайд با ديگر عواملهاي هزينه - که در همين فصل شرح داده شده اند - ترکيب شود. نكته قابل توجه اين است که متن باز تلاش می کند که فرصتی برای رسیدن به يك رویکرد تقریباً كامل با معقول ترین هزينه، فراهم کند. تحلیلهای تجاری شما تعیین خواهد کرد که آيا رویکرد جدید ارزش سرمایه گذاری دارد یا خیر.

۶-۳- فراهم کردن لینوکس و نرم افزارهای متن باز

شرکت شما، معمولاً هنگام در اختیار گرفتن محصولات نرم افزاری سنتی فرآيند جامع و فرآگيری برای ایجاد ارتباط با عرضه کنندگان و تولید کنندگان راه حلها ایجاد می کند. شما با توجه به حجم نرم افزارها در مورد هزینه های مجوز آنها مذاکره می کنید و تعداد محدودی یا حتی يك تأمین کننده را برای اين کار انتخاب می کنید تا هزینه ها را به حداقل برسانيد. فرآيند در اختیار گرفتن لینوکس، آپاچی و یا نرم افزارهای متن باز دیگر کلّاً با یکدیگر متفاوت است. مثالی از در اختیار گرفتن سامانه عامل لینوکس را در نظر بگيريد. موارد ارائه شده معمول ترین راههای داشتن يك نسخه از لینوکس هستند:

- خرید یک جعبه حاوی رسانه نصب کننده لینوکس از یک فروشنده محلی
- دریافت تصاویر لوجهای فشرده آن از اینترنت
- استفاده از لوح فشردهای که به همراه یک کتاب یا محصول عرضه می‌شود
- استفاده از نرم‌افزاری که در یک سامانه رایانه‌ای نصب شده یا همراه تجهیزات جانبی رایانه است

روشهای مشابهی برای در اختیار گرفتن نرم‌افزارهای متن‌باز دیگر وجود دارد. در برخی موارد عرضه کنندگان نرم‌افزار، برنامه‌های کمکی متن‌باز، ابزارها و کاربردهای اضافی را با نرم‌افزار خود همراه می‌کنند.

عاملان خرید در سازمان شما، اغلب برای در اختیار گرفتن و داشت حساب مجوز نرم‌افزارهایی که خریداری می‌شوند، آموزش داده می‌شوند. این احتمال وجود دارد که آنها بخواهند همین فرآیندها را روی نرم‌افزارهای متن‌باز اعمال کنند. به هر حال مهم است که مأموران خرید شما به گونه‌ای آموزش دیده باشد که تنها نرم‌افزارهای متن‌بازی را خریداری کند که ارزش صرف هزینه را دارند. در مواردی که عرضه کنندگان سخت‌افزار یا نرم‌افزار، نرم‌افزارهای متن‌باز نشان دار را با محصول خود همراه می‌کنند، دقت کنید که ممکن است این عرضه کننده هزینه مجوز این نرم‌افزار را روی قیمت محصول خود کشیده باشد. همراه سازی نرم‌افزارهای متن‌باز نشان دار با یک محصول به قراردادهای تجاری مربوط می‌گردد که بین دو شرکت بسته می‌شود و اجازه این کار را به آنها می‌دهد.

در اکثر موارد عرضه کنندگان سخت‌افزار، نرم‌افزار، مؤلفه‌های متن‌بازی را همراه محصول خود می‌کنند که در جای دیگر بدون هیچ هزینه‌ای موجود باشند. به هر حال ممکن است در مواردی تسهیلاتی که همراه سازی فراهم می‌کند، ارزش پرداخت هزینه اضافه را داشته باشد. نکته اینجاست که باید متوجه باشید به ازای پولی که پرداخت می‌کنید، چه دریافت می‌کنید. امکان بدست آوردن نرم‌افزارهای متن‌باز از منابع دیگر یا نسخه‌برداری به تعداد دلخواه همیشه ممکن است. در اینجا نیازی به ردیابی مجوز نرم‌افزارها نیست.

قراردادها

اگر شما عرضه کننده سخت‌افزار یا نرم‌افزار هستید، طبیعی است که از عرضه کننده‌ای که محصول وی را با محصول خود همراه می‌کنید، در مقابل صحت محصول وی مطالبه خصمانت نمائید. اما وقتی حرف از نرم‌افزارهای متن‌باز به میان می‌آید، قواعد تعییر می‌کند. از آنجایی که نرم‌افزارهای متن‌باز توسط مجموعه‌ای از افراد مختلف نوشته می‌شوند و بدون اغراق صدها مالک حق نشرهای مختلف بر روی آن دارند، کسی برای شما چنین تضمینی نمی‌کند.

به خاطر داشته باشید در هر مذکورهای با یک عرضه کننده تجاری، شما تنها در مورد هزینه مجوز نرم‌افزار مذکوره نکرده بلکه اغلب برای پشتیبانی، خدمات و مواردی از این قبیل مذکوره می‌کنید. همچنین به خاطر داشته باشید که این محیط رقابتی است و گزینه‌های زیادی برای انتخاب وجود دارد.

۶- تغییردادن نرم افزار متن باز

بسیاری از تحلیل گران صنعتی، تغییر نرم افزار متن باز را توصیه نمی کنند و پیشنهاد دارند که با این قبیل نرم افزارها همانند دیگر نرم افزارهای اختصاصی در شرکت رفتار شود. این توصیه از این واقعیت نشأت می گیرد که با تغییر دادن نرم افزار طبق نیازها و استفاده های خاص، باید از پشتیبانی که برای نرم افزار جدید انتظار می رود، چشم پوشی کرد. شاید برای شرکتهای کوچکتر با پرسنل کمتر در فناوری اطلاعات، این دلیل قانع کننده ای برای تغییر ندادن نرم افزارهای متن باز باشد، ولی برای شرکتهای بزرگ که با جامعه متن باز تعامل دارند، تغییر نرم افزار می تواند کمک خوبی برای کاهش هزینه ها باشد.

تغییر یک نرم افزار متن باز بهتر است همیشه با هماهنگی نگهدارندگان آن در جامعه متن باز انجام شود. توانایی تغییر نرم افزار مزایایی را هم برای شرکتهای بزرگ و هم برای تولیدکنندگان نرم افزار فراهم می کند:

- تأثیر در جهت پیشرفت محصول نرم افزاری

سرمایه گذاری در عملی کردن اولویتها با زمانبندی مطلوب

- داشتن دید عمیق و کلی از داخل سازمان برای برنامه ریزیهای آینده

توجه کنید که نگهدارنده کنترل نرم افزار را در دست دارد. مشارکت پیوسته باعث تأثیر گذاری بر وی می شود. هنگام تحلیل هزینه های لینوکس و متن باز، مقایسه هزینه انتظار برای عملی شدن اولویتها با هزینه های سرمایه گذاری و مشارکت مستقیم خودتان در احقاق اولویتها در نظر بگیرید. اگر مایل نیستید که به انجمن پیوندید و یا قادر نیستید که این کار را انجام دهید و نمی توانید در آینده هر محصول نرم افزاری شرکت کنید، آن وقت است که باید حق را به تحلیل گران بدھید؛ بهتر است که از هر تغییر در سامانه متن باز بپرهیزید و فقط زمانی از آن استفاده کنید که بسته بندی شده و توسط یک عرضه کننده تجاری حمایت می شود.

۶- جمع بندی

شما اکنون باید متوجه شده باشید که لینوکس و متن باز چگونه هزینه‌های یک راه حل را در سازمان را تحت تاثیر قرار می‌دهند. در اکثر موارد می‌توانید از مزایای عمدۀ آن بهره برید و هزینه‌ها را کاهش دهید. به هر حال شما متوجه این مسئله هم شدید که به کارگیری متن باز به این معنی نیست که هزینه‌ها به صفر می‌رسد و متحمل هیچ هزینه‌ای نخواهید بود. برای نرم‌افزارهای متن باز هیچ حق الامتیازی مربوط به مجوزها نخواهیم داشت و نیاز نیست برای آنها مبلغی پرداخت شود. حتی اگر با مؤلفه‌های دیگری همراه شوند. ولی در عین حال برای نگهداری راه حل خود باید هزینه پرداخت کنید. فرآیندهای مذاکره برای قرارداد ممکن است نیاز به تجدید نظر داشته باشند. نهایتاً اینکه شما می‌توانید کنترل بیشتری روی هزینه‌ها از طریق تصمیم‌گیریهای معقولانه و روشن در مورد تغییر دادن نرم‌افزار و برگرداندن آن به جامعه و بهره‌گیری از مزایای رویکردی که بیشترین نیازهای تجاری شما را برآورده می‌کند، داشته باشید. یکی از عناصر اصلی در پائین نگهداشتن هزینه‌ها در اکوسیستم، استانداردها هستند. فصل بعدی استانداردهای خاص محیط لینوکس را بررسی خواهد کرد. ارتقاء این استانداردها، رشد اکوسیستم را تسريع نموده و هزینه‌ها را کاهش خواهد داد.

فصل ۷

استانداردها - یک لینوکس

اکنون شما اتوموبیل خود را تکمیل کرده‌اید. این اتوموبیل را با ساختن موتور آن آغاز کردید. سپس امکانات مورد نیاز را به آن افزودید. اکنون از هزینه‌های آتی اتوموبیل خود نیز آگاهید. حال تصور کنید اتوموبیل افراد مختلف آنقدر متفاوت باشند که نتوانند در یک جاده حرکت کنند و از یک سوخت استفاده کنند. در این صورت استفاده از اتوموبیل در انحصار افرادی خواهد بود که توانایی ایجاد جاده‌ها و بزرگراه‌های اتوموبیل خود را داشته باشند. در یک اتوموبیل، داشتن امکانات مناسب با نیازهای مشتریان به اندازه ملزم بودن به رعایت استانداردها مهم است.

در این فصل جزئیات استانداردهای تعیین شده در حوزه لینوکس و متن‌باز را بررسی خواهیم کرد. در نظر داشته باشید که فقط استانداردهای جدید و مخصوص لینوکس مورد بحث قرار خواهند گرفت. همچنین جنبه‌های مختلف یک استاندارد را که باعث قابل انتقال بودن نرم‌افزار می‌شوند، بررسی خواهیم کرد. به همین دلیل اهداف این فصل درک موارد زیر است:

- چرا به استانداردهای لینوکس و متن‌باز احتیاج داریم؟
- استانداردهای کلیدی و نحوه کار آنها
- تطابق و آزمایش کردن
- جایی که از استانداردها استفاده نمی‌شود

اینها اطلاعات لازم برای استفاده مؤثر از مجموعه استانداردهای متن‌باز را در اختیار شما قرار می‌دهند. بدین ترتیب می‌توانید از تمامی مزایای یک محیط باز استفاده کنید.

۷-۱- چرا استاندارد؟

همان گونه که در فصل قبل فرا گرفتید، توزیع لینوکس ترکیبی است از هسته لینوکس، پیمانه‌ها، وصله‌ها، ابزارهای GNU و مجموعه‌های از برنامه‌های دیگر. کار فروشنده‌گان توزیعهای لینوکس را در زمینهٔ بسته‌بندی لینوکس باید تقسیم کرد. اکنون نصب و استفاده از لینوکس امری بسیار آسان شده است. اما این روح همکاری و نیاز به نوآوری باعث ایجاد تفاوت‌های زیادی میان توزیعهای مختلف شده است. از این تفاوت‌ها هیچ کس سودی نمی‌برد. در اینجا به چند نمونه از تصمیماتی که فروشنده‌گان توزیعها باید بگیرند، اشاره می‌شود:

- نسخه هسته
- وصله‌ها
- پیمانه‌ها، درایورها و نسخه‌ها
- موقعیت پرونده‌ها
- نسخه‌های بسته‌های نرم‌افزاری

تقریباً همه افراد درگیر در صنعت لینوکس (کاربران، فروشنده‌گان توزیعها و فروشنده‌گان برنامه‌های کاربردی) به این نتیجه رسیده‌اند که این تفاوت‌ها باعث تأخیر در به کارگیری لینوکس می‌شوند. در نتیجه فروشنده‌گان بزرگ لینوکس به همراه فروشنده‌گان سخت‌افزار و نرم‌افزار و

برنامه‌نویسان سرشناس متن باز با یکدیگر متحد شدند تا مجموعه استانداردهایی برای لینوکس تعریف کنند. این استانداردها توسط گروه استانداردهای آزاد^۱ مدیریت می‌شوند. مسلماً باید مزایایی در تعیین این استانداردها وجود داشته باشد. قبل از اینکه به جزئیات این استانداردها نگاهی بیاندازیم، مزایای عمدۀ این استانداردها برای افراد دخیل در این صنعت را بر می‌شماریم.

- کاربران- کاربران نهایی اغلب شرکتهای متوسط را بزرگ با حضور جهانی هستند. استانداردهای لینوکس به کاربران این امکان را می‌دهد که نیازهای محلی خود را در انتخاب توزیع لینوکس دخیل کنند و در عین حال از سازگاری آن با نرم‌افزارهای استاندارد مطمئن باشند. مزیت دیگر استانداردها برای کاربران این است که حداقل نرم‌افزارهای موجود را در توزیعهای منتخب خود استفاده کنند.
- فروشنده‌گان توزیعها- سرمایه لازم برای ایجاد یک توزیع لینوکس کم نیست. هر چه استانداردها بیشتر باشد، تشابه بین توزیعهای بیشتر می‌شود و در نتیجه هزینه کمتری در برخواهد داشت. همچنین استانداردها باعث بزرگ شدن بازار می‌شود. اینترنت را در نظر بگیرید. اگر استانداردها در این زمینه وجود نداشتند، اندازه اینترنت بسیار کوچک‌تر می‌بود و ما شاهد تغییرات صنعتی اواخر دهۀ ۱۹۹۰ نمی‌بودیم. فروشنده‌گان توزیعها از دو جنبه سود می‌برند: ۱) حفظ سازگاری بین نسخه‌های مختلف؛ ۲) داشتن تعریف مشخصی از محل اتمام لایۀ اشتراک و آغاز لایۀ منحصر به فرد ارزش افزوده.
- فروشنده‌گان نرم‌افزار- تجربه تولید نرم‌افزار برای گونه‌های مختلف یونیکس اثبات کرده است که این امر بسیار پرهزینه است. فروشنده‌گان نرم‌افزار مجبور هستند که بین هزینه پشتیبانی نسخه‌های مختلف و بزرگی بازار تعادل برقرار کنند. استانداردها این امکان را می‌دهد که فروشنده‌گان نرم‌افزار یک نسخه از هر نرم‌افزار را تولید کنند و آنرا تحت بسترهای مختلف پشتیبانی کنند.

لينوکس و نرم‌افزارهای متن باز امروزه در حد خوبی رشد نموده‌اند و امروزه برای اطمینان از حفظ رشد این صنعت نیازمند داشتن استانداردهایی هستیم که بر نوآوری متتمرکز باشند.

۲-۷- گروه استانداردهای آزاد

گروه استانداردهای آزاد (www.freestandards.org) سازمانی غیرانتفاعی است که اولین وظیفه آن گسترش استفاده و مرغوبیت تکنولوژیهای متن باز، از طریق استانداردها، می‌باشد. محصول طبیعی کار آنها این خواهد بود که لینوکس به سمت مرحله بعدی پشت بازار یعنی بستر شرکتی برای برنامه‌های تجاری حرکت خواهد کرد. گروه‌های کاری اصلی این سازمان به شرح زیر می‌باشند:

^۱ Free Standards Group

- پایه استانداردهای لینوکس (LSB) مجموعه‌ای از استانداردها و آزمایشها برای رابطها می‌باشد. این استانداردها باعث قابل انتقال بودن برنامه‌ها از توزیعی به توزیعی دیگر صرفنظر از نوع و نسخه آن می‌شوند.
- طرح بین‌المللی‌سازی لینوکس (Li18nux) – این استاندارد بستری برای بومی‌سازی زبانی و فرهنگی، توزیعها و برنامه‌ها فراهم می‌کند.
- استاندارد سلسله مراتب سامانه‌پرونده (FSH) – این استاندارد، مدل ثابتی برای محل قرار دادن پرونده‌ها در یونیکس و امروزه لینوکس، ارائه می‌کند. FSH یکی از استانداردهای کلیدی است که LSB به آن وابسته است. استاندارد FSH به همراه LSB بررسی خواهد شد.

گروه استانداردهای آزاد همه استانداردها را به تنها‌یی تعیین نمی‌کند، بلکه با طراح استاندارد دیگری برای لینوکس نیز همکاری دارد: گروه X Desktop. گروه X Desktop استانداردهایی فعالیت می‌کند که همکاری و قابلیت جابجایی بین محیط‌های رومیزی لینوکس و یونیکس را تشریح می‌کند. اکنون می‌دانید که دو محیط رومیزی اصلی برای لینوکس وجود دارد: GNOME و KDE. گروه استانداردهای آزاد همکاری نزدیکی نیز با گروه XFree86 (به فصل ۴ مراجعه کنید) دارد.

گروه غیرانتفاعی استانداردهای آزاد توسط شرکتهای بزرگ این صنعت تأسیس شده است. این شرکتها به خوبی مزایای وجود داشتن این استانداردها را می‌دانند. هر کاربر یا تولیدکننده نرم‌افزار و سخت‌افزاری تشویق به همکاری در ایجاد این استانداردها می‌شود. استانداردها وقتی موفق هستند که نیازهای کاربران را برآورده کنند و به طور گسترده‌ای در جامعه به کار گرفته شوند.

۷-۳-۳- پایه استانداردهای لینوکس (LSB)

LSB (www.linuxbase.org) مهمترین استانداردی است که باعث ازدواج موفق توزیعهای لینوکس و برنامه‌های لینوکس می‌شود. این استاندارد از اجزاء زیر تشکیل شده است:

- مشخصات دودویی رابط (مشخصات دودویی الزام می‌کند که برنامه‌ها نباید موقع انتقال از یک سامانه به یک سامانه دیگر نیازمند ترجیم مجدد باشند). که باعث سازگاری نرم‌افزارها با تمامی توزیعهای لینوکس تهیی شده می‌شود.
- مجموعه‌ای از آزمایشها بر روی توزیعها و برنامه‌ها برای اطمینان از تطابق با LSB.
- ایجاد محیطی برای گسترش نرم‌افزارهای سازگار با LSB.
- نمونه‌های پیاده شده LSB، برای اجرا و آزمایش برنامه‌ها در یک محیط LSB خالص.

با به کارگیری استانداردها و ابزارها فروشنده‌گان می‌توانند توزیعهای لینوکس منطبق با LSB و فروشنده‌گان نرم‌افزارهای منطبق با LSB تولید کنند.

تولیدکننده‌گان نرم‌افزار با تمرکز بر مجموعه رابطه‌ای مشترک می‌توانند برنامه خود را مطابق LSB تولید و آزمایش کنند و مطمئن باشند که این برنامه بر روی هر توزیع منطبق با LSB اجرا می‌شود. شرکتهایی هم که تیم برنامه‌نویسی دارند باید همه برنامه‌های خود را مطابق با LSB

بنویسند. بنابراین قبل از تهیه توزیع لینوکس یا هر برنامه‌ای از تطابق آن با آخرین نسخه LSB اطمینان حاصل کنید. بدین طریق سرمایه‌گذاری شما ماندگاری بیشتری خواهد داشت.

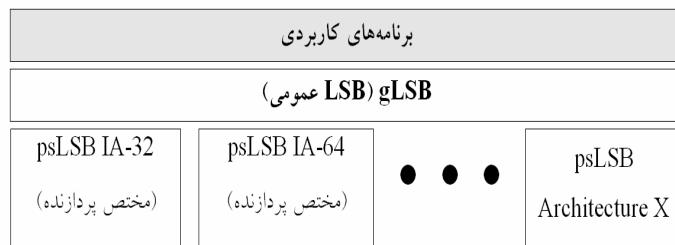
به LSB می‌توان به عنوان جمع‌آوری کننده مجموعه استانداردهای در حال استفاده در صنعت و مورد تأیید جامعه برنامه‌نویسان لینوکس، نگاه کرد. یکی از اهداف مشخص LSB این است که استانداردهای موجود را مستندسازی کند نه اینکه استاندارد جدیدی اختراع کند. LSB اختراع مجدد نیست بلکه گسترش چیزی است که قبلاً اختراع شده است. به همین دلیل این استاندارد به راحتی مورد قبول تولیدکنندگان برنامه کاربردی و توزیعهای لینوکس قرار گرفته است و تطابق آنها با این استاندارد مشکل نیست.

از آنجایی که LSB یک مشخصات دودویی است، لازم است که به دو جزء اصلی تقسیم شود: عمومی و بسته به نوع پردازنده؛ عمومی برای چیزهایی که مستقل از دستگاه هستند و بسته به نوع پردازنده فقط برای چیزهایی که باید وابسته به نوع دستگاه باشد. شکل ۱-۷ نشان می‌دهد که چگونه با تلفیق این دو بخش می‌توان یک محیط منطبق با LSB برای یک معماری خاص طراحی کرد.

در اینجا قصد نداریم کل استاندارد LSB را بازخوانی کنیم. یک نسخه از این استاندارد را می‌توانید از سایت www.linuxbase.org/spec دریافت کنید. خالی از لطف نخواهد بود که نگاهی سطح بالا بر این استاندارد و نحوه کنار هم آمدن اجزاء مختلف بیاندازیم.

۴-۷- اجزای توزیعهای LSB

LSB مجموعه رابطهایی را که برنامه‌ها باید داشته باشند تعریف می‌کند. شکل ۲-۷ اجزاء استاندارد شده یک توزیع لینوکس منطبق با LSB را نشان می‌دهد.



شکل ۱-۷: پشتیبانی LSB از معماریهای پردازنده‌ها



شکل ۲-۷: توزیع مطابق با LSB

در جدول ۱-۷ اجزای مهم LSB عمومی (gLSB) در یک توزیع لینوکس و تأثیر آنها بر برنامه‌ها را شرح می‌دهد:

جدول ۱-۷ اجزای gLSB

جزء	شرح
رابط هسته لینوکس	گروههای توسعه هسته باید در نوآوریها و گسترش تواناییهای هسته آزاد باشند. LSB ساخته شد تا نوآوری را گسترش دهد. با استفاده از رابط POSIX (استاندارد بخوبی جاافتاده در زمینه رابطهای یونیکس) در عین تشویق نوآوری، سازگاری برنامه‌ها را تضمین می‌کند. مفهوم این کار این است که اجازه داده می‌شود که ساختار کنونی یونیکس نیز با LSB تطبیق کند.
چارچوب بسته‌بندی	بسته بندی به فرآیند ترکیب مجموعه پروندهای تشکیل دهنده یک برنامه، برای استفاده آسان تر، گفته می‌شود. با تعیین چارچوب بسته بندی، فروشنده‌ها می‌توانند از مجموعه ابزار موجود برای پشتیبانی این چارچوب انتخاب کنند.
پیوندسازی پویا	پیوندسازی پویا قالب دودویی و فیلدی‌های مورد نیاز برنامه را مشخص می‌کند. پیوندسازی پویا این امکان را فراهم می‌آورد که برنامه‌های زیادی بتوانند از کتابخانه‌های توابع مشترکی استفاده کنند.

جزء LSB	شرح
محیط اجرا	<p>محیط اجرا بر استاندارد FSH و امنیت اجرا تمرکز یافته است. همان‌گونه که قبلاً اشاره شد، استاندارد FSH از اجزای اصلی LSB می‌باشد. با اطمینان از هماهنگی موقعیت قرارگیری پروندها در سامانه قابلیت پیش‌بینی برنامه‌ها را افزایش می‌دهد.</p>
راه اندازی سامانه	<p>راه اندازی سامانه وابستگی اکثر برنامه‌ها به یک وضعیت شناخته شده در هنگام راه اندازی سامانه را به رسمیت می‌شناسد. این قسمت از استاندارد، اطمینان حاصل می‌کند که برنامه‌ها محیطی در اختیار دارند که می‌توانند روی آن حساب کنند، مانند دستنوشته‌ها و پیکربندیها.</p>
پوسته استاندارد	<p>در طول سالها تعداد زیادی پوسته (تفسری) که دستورات تایپ شده در خط فرمان را اجرا می‌کند) طراحی شده‌اند. این قسمت از LSB استاندارد POSIX را به رسمیت می‌شناسد و در عین حال واقعیت‌های پراستفاده‌ترین پوسته در لینوکس را نیز برمی‌شمارد.</p>
قالیبایی پرونده‌های اشیاء	<p>وقتی که مترجمی (کامپایلر) برنامه اجرایی را با استفاده از یک کد منبع می‌سازد، آن پرونده را باید براساس قاعده کاملاً مشخصی روی دیسک بنویسد. اگر استاندارد مشخصی برای این قاعده وجود داشته باشد، تمامی برنامه‌ها می‌توانند در حافظه بارگذاری شده اجرا شوند.</p>
کتابخانه‌ها	<p>در نمودار ملاحظه می‌کنید که سه نوع کتابخانه وجود دارد: پایه، گرافیک و کمکی. در تئوری تمامی برنامه‌های بارگذاری شده در سامانه حداقل به یکی از کتابخانه‌های هریک از این سه دسته وابسته است. مشخص کردن اینکه کدامیک از این کتابخانه‌ها باید در دسترس برنامه‌ها باشند باعث می‌شود برنامه‌ها با اطمینان بیشتری اجرا شوند.</p>
دستورات و برنامه‌های کمکی	<p>تمامی سامانه‌های یونیکس و لینوکس صدها دستور در اختیار مدیران سامانه و کاربران قرار می‌دهند. این بخش از استاندارد دستورهایی را که باید در اختیار کاربران باشد تصریح می‌کند. به این ترتیب برنامه‌هایی نیز که وابسته به اسکریپتها هستند با اطمینان بیشتری اجرا می‌شوند.</p>

جزء	شرح
کاربران و گروهها	بازهم با به کارگیری اهرم استاندارد POSIX نام کاربرها و گروهها نیز از استانداردها تعیت می‌کنند.

همان‌گونه که گفته شد LSB اجزاء وابسته به معماری را نیز مشخص می‌کند. همان‌گونه که در فصل دوم اشاره کردم، از آنجایی که اکثر برنامه‌ها تحت معماری Intel IA-32 نوشته می‌شوند، همهٔ طراحیهای اولیه از اینجا آغاز می‌شود. LSB هم مستثنی نیست. در زمان نوشتمن این کتاب بخش وابسته به پردازنده LSB در زمینه IA-32 تعریف شده است و پیش‌نویس‌های IA-64 در حال تهیه هستند. در جدول ۲-۷ اجزاء LSB برای IA-32 آورده شده است.

جدول ۲-۷ اجزای وابسته به معماری LSB

جزئی	شرح
اطلاعات سطح پایین سامانه‌ای	در این بخش رابط بسیار سطح پایین معماری IA-32 شرح داده شده است. این بخش مطمئن می‌شود که کامپایلرهای استانداردهای مورد نیاز را برای ساختن برنامه‌هایی که بخوبی اجرا می‌شوند دارند.
قالب پرونده هدف	همانند مشخصات gLSB در این زمینه، مطمئن می‌شود که کامپایلرهای پروندهای هدف را به درستی می‌نویسند.
بارگذاری برنامه‌ها و پیوندسازی پویا	فرآیند اجرای برنامه شامل دو بخش بارگذاری برنامه در حافظه و زمانبندی (پردازنده) برای اجرا آن تشکیل شده است. همان‌گونه که در gLSB ذکر شده است، در فاز بارگذاری، برنامه باید با تمامی کتابخانه‌های مورد نیاز پیوند برقرار کند. این بخش از استاندارد مطمئن می‌شود که فرآیندهای بارگذاری، پیوندسازی، زمانبندی و جاددادن یک برنامه اجرایی در حافظه از قاعده درستی پیروی می‌کنند.
کتابخانه‌های پایه	gLSB مجموعه‌ای از کتابخانه‌های پایه را معرفی می‌کند. تعدادی کتابخانه هم مخصوص معماری IA-32 وجود دارد که در اینجا معرفی می‌شوند.

استانداردی فرآگیر و ابزاری است که انتظارات یک برنامه از یک سامانه منطبق با آن را شرح می‌دهد. همچنین قواعدی وجود دارد که نرم‌افزارها برای تطابق با LSB باید آنها را رعایت کنند.

۵-۷- برنامه‌های منطبق بر LSB

اطمینان از قابل اجرا بودن نرم‌افزارهای در حال طراحی، تحت همه لینوکسها، چه این برنامه برای فروش باشد، چه برای استفاده داخل سازمان برد بزرگی است. با این کار شما این امکان را در اختیار کاربران قرار می‌دهید که نوع لینوکس خود را از مجموعه لینوکس‌های موجود انتخاب کنند، نه اینکه مجبور به استفاده از توزیع خاصی باشند. اگر کاربری به هر دلیل مجبور به تغییر توزیع لینوکس خود شود، سرمایه‌ای را که صرف تهیه نرم‌افزار شما کرده است از دست نمی‌دهد. ساده‌ترین تصور از انطباق نرم‌افزار با LSB این است که می‌تواند بر روی هر توزیع منطبق با LSB اجرا شود. چنین نرم‌افزاری نباید به بخش‌های اضافه برخی توزیعها که در LSB نیامده‌اند، وابسته باشد. همچنین اگر چنین برنامه‌ای لازم است با برنامه دیگری ترکیب شود یا وابستگی به آن داشته باشد، آن برنامه نیز باید منطبق بر LSB باشد.

البته برنامه‌شما باید از تمامی قواعد ذکر شده در LSB پیروی کند. در ادامه آزمایش تطابق را بررسی خواهیم کرد که به شما اطمینان می‌دهد نرم‌افزارتان با استاندارد LSB منطبق است.

۶- آینده LSB

LSB تاکنون تلاش کرده است مشخصات تمامی آنچه را که مربوط به لینوکس می‌شود، پوشش دهد. آینده‌های LSB یکی از کمیته‌های فرعی گروه کاری LSB است. هدف آن گسترش محدوده‌های کاری LSB است. اگر مایلید LSB را گسترش دهید باید با این گروه کار کنید.

۷- بین‌المللی‌سازی لینوکس

دومین گروه کاری اصلی گروه استانداردهای آزاد، برنامه بین‌المللی‌سازی لینوکس^۱ (www.li18nux.org) است. این گروه گروهی از API‌های اصلی لینوکس را استاندارد می‌کند به گونه‌ای که برنامه‌های بومی شده به راحتی بر روی لینوکس‌های LSB اجرا شوند.

فروشنده‌گان توزیعهای لینوکس از استانداردهای^۲ li18nux بهره می‌برند بدین ترتیب که اطمینان دارند که توزیع آنها در هر نقطه‌ای از دنیا کار می‌کند و می‌تواند برنامه‌های بومی شده هر منطقه را اجرا کند.

شرکتهای بزرگ و تولیدکنندگان نرم‌افزار نیز با استفاده از این استاندارد، می‌توانند نرم‌افزار خود را مناسب با مشخصات بومی هر منطقه‌ای طراحی کنند.

^۱ Linux Internationalization Initiative

^۲ عدد ۱۸ در li18nux نماینده ۱۸ حرف بین n و o دز کلمه internationalization است.

در اینجا به شرح چند اصطلاح می‌پردازیم:

- بین‌المللی‌سازی – فرآیند طراحی یا اصلاح یک برنامه یا بستر نرم‌افزاری، به گونه‌ای که قابل بومی شدن باشد.
 - بومی سازی فرآیند پشتیبانی از مشخصات فرهنگی از قبیل زبان، واحد پول در یک برنامه. پس از اینکه برنامه‌ای بین‌المللی شد، می‌توان آن را برای هر منطقه جغرافیایی بومی سازی کرد.
 - جهانی سازی – ترکیب بین‌المللی‌سازی و بومی‌سازی اگر قصد داردید برنامه‌ای برای کاربرد داخل سازمان یا فروش تجاری بنویسید، باید خود را ملزم به بین‌المللی‌سازی آن کنید. با این کار تضمین می‌کنید که برنامه شما در صورت نیاز برای هر کشوری قابل بومی سازی است. برخی شرکتها وجود دارند که متخصص ترجمه و بومی‌سازی هستند و در صورت نیاز به بومی‌سازی برنامه، می‌توان از متخصص آنها بهره برد.
- تعدادی API در اختیار شما قرار می‌دهد که می‌توانند با مجموعه کاراکترهای لاتین، آسیایی و تعداد زیادی از کشورهای دیگر کار کنند. همچنین ابزارهایی برای آزمایش انطباق برنامه با li18nux در اختیار قرار می‌دهد. با li18nux بدون نیاز به تغییر کد برنامه خود می‌توانید قابلیت پشتیبانی از هر زبانی را به آن اضافه کنید.

۸-۷- آزمایش و تطابق

گروه استانداردهای آزاد از هر فرصتی برای مستندسازی آنچه که موجود است استفاده می‌کند. جنبه‌های آزمایش و انطباق نیز از این قاعده مستثنی نیستند. در دنیای یونیکس، "گروه باز" برای اطمینان از تطابق برنامه‌ها با استانداردها. آزمایشهای گسترده‌ای برروی آنها انجام می‌دهد. گروه استانداردهای آزاد با گروه باز و سایر گروههای حرفه‌ای در زمینه لینوکس تعامل دارد و بهجای اینکه از اول شروع کند، از تجربه آنها برای طراحی آزمایشهای خود بهره می‌برد. مجموعه آزمایشهای تطابق با LSB به ۴ دسته تقسیم می‌شوند:

- آزمایش توزیعهای لینوکس – این بخش از آزمایش‌ها مخصوص فروشنده‌گان توزیعهای لینوکس است. ممکن است شما بخواهید لینوکس مورد استفاده سازمان خود را در داخل سازمان طراحی کنید. در حالت هم باید توزیع شما مطابق با LSB طراحی شده باشد تا بتوانید برنامه‌های مطابق با LSB را روی آن اجرا کنید.
- آزمایش برنامه‌ها این مهمترین آزمایش برای تولیدکنندگان نرم‌افزار و خریداران شرکتی می‌باشد. حتی اگر برنامه‌های شما برای استفاده داخل سازمان نوشته می‌شوند باید مطابق با LSB نوشته شوند. این کار تضمین می‌کند که برنامه شما برروی هر توزیع مطابق با LSB کار می‌کند.

¹ Open Group

- آزمایش محیط ساخت – این آزمایش هم مخصوص تولیدکنندگان نرمافزار است. این آزمایش محیط ساختن (نوشتن) برنامه را از نظر داشتن تمامی امکانات مورد نیاز LSB، تحت آزمایش قرار می‌دهد.
- آزمایش نمونه پیاده‌سازی شده – این آزمایش، روش نهایی آزمایش تطابق با LSB است. در این روش یک محیط اجرای برنامه ارایه می‌شود که نوعی سامانه [عامل] حداقل و به طور خالص LSB می‌باشد. تولیدکنندگان نرمافزار می‌توانند برنامه‌های خود را تحت این سامانه آزمایش کنند.

li18nux تعدادی آزمایش تطابق دیگر نیز دارد که می‌توانید آنها را در وب‌گاه اینترنتی آن ببایدید. مجموعه تستهای li18nux به دو صورت وجود دارند: تعاملی که اجزای قابل دیدن برنامه را آزمایش می‌کنند و خودکار که پیاده‌سازی درست رابطه‌ای بین‌المللی‌سازی را آزمایش می‌کنند. در زمان نوشتمن این کتاب، گروه استانداردهای آزاد در حال آغاز برنامه‌ای برای دادن تاییدیه به فروشنده‌گان توزیعها و نرمافزارها بود که بتوان آن را به صورت برچسبی ببروی بسته‌بندی محصول چسباند. برای اطلاع از این برنامه باید به وب‌گاه آنها مراجعه کنید.

۹-۷- توزیعهای تخصصی لینوکس

استانداردهایی که توسط گروه استانداردهای آزاد توسعه یافته‌اند بیشتر بر توزیعهای لینوکس قابل استفاده برای عموم متوجه شده‌اند. نباید فراموش کرد که یکی از قابلیتهای بزرگ لینوکس قابلیت سفارشی کردن آن (تغییر متناسب با نیاز) می‌باشد. به همین دلیل کاربردهای بسیار زیادی از لینوکس وجود دارد (SetupBox، PDA)؛ استفاده‌های توکار و در این کاربردها چنان لینوکس را تغییر می‌دهند که دیگر نمی‌توان آن را استاندارد دانست. خوشبختانه با فراگیر شدن این کاربردها تلاشهایی برای استاندارد کردن این محیطها آغاز شده است.

۱۰-۷- جمع بندی

در این بخش با استانداردهای دنیای فراگیر لینوکس آشنا شدید. با استفاده از توزیعها و برنامه‌های مطابق با LSB مطمئن باشید که هیچ‌گاه در دام یک فروشنده توزیع یا نرمافزار گرفتار نخواهد شد. در زمینه شرکتهای بزرگ خودکفا از نظر نرمافزار و تولیدکنندگان نرمافزار نیز پیروی از li18nux آنها را قادر می‌سازد محصولات خود را با کمترین هزینه در بازارهای کشورهای دیگر عرضه کنند.

فصل هشتم

عملیات: به کارگیری لینوکس و متن باز

برای تحقق پیاده‌سازی لینوکس در یک محیط، باید آن را یکی از ملزمات انجام کارهای روزمره قرار داد. در اکثر کسب و کارها لازم است برنامه‌ای برای به کارگیری لینوکس در ساختار موجود تدوین کرد. کسانی که قصد دارند از محیط ویندوز یا یونیکس به لینوکس مهاجرت کنند، در ابتدا باید مهاجرت را درست درک کرده باشند و برنامه‌ای برای این مهاجرت داشته باشند. در ادامه این فصل در زمینه پشتیبانی و آموزش لینوکس بحث خواهد شد.

در این فصل موارد عملی مورد بحث قرار خواهند گرفت. از آنجایی که ساختار سازمانها با هم فرق می‌کند، این فصل را باید به عنوان ابزاری کمکی برای تصمیم‌گیری در زمینه به کارگیری لینوکس تلقی کرد. در اینجا یک مورد خاص مهاجرت از یونیکس یا ویندوز یا هر سامانه عامل دیگری به لینوکس را بررسی نمی‌کنیم، بلکه ملاحظات مهاجرت به لینوکس معرفی خواهند شد. قابل ذکر است در مواردی مهاجرت به لینوکس ممکن است صلاح نباشد و نباید انجام شود.

۱-۸- پیاده سازی

قبل از پیاده‌سازی لینوکس در یک سازمان، باید طرح جامع این کار از قبل طرح‌بیزی شده باشد و مشخص باشد که کدام برنامه‌ها را باید تحت لینوکس پیاده‌سازی کرد. در اکثر موارد باید لینوکس را در کنار ساختار دیگری نصب کرد. حتی اگر قصد داشته باشید در تمام زیرساخت سازمان خود از لینوکس استفاده کنید، رسیدن به این هدف ممکن است ماه‌ها یا حتی سال‌ها طول بکشد.

برای پیاده‌سازی لینوکس در یک سازمان در ابتدا باید مشخص باشد که از کدام توزیع آن و با چه پیکربندی قرار است استفاده شود. همان گونه که در فصل مربوط به توزیعها ذکر شد، در اغلب موارد باید یک توزیع سفارشی برای خود تهیه کنید. بسته نرم‌افزاری لینوکس سفارشی باید به گونه‌ای باشد که از عهده پشتیبانی آن برآید. در قدم بعد باید برنامه‌ها و ابزار تجاری یا متن باز موجود برای لینوکس را بررسی کنید و آنچه را که مناسب نیازهای شماست بیابید. پس از یافتن برنامه‌ها بعد آنها را به خوبی آزمایش کنید و از کارایی آن اطمینان حاصل کنید.

اکثر فروشنده‌گان توزیعهای لینوکس خدماتی را نیز ارائه می‌دهند که از طریق آنها می‌توان لینوکس تهیه شده را به روز کرد. برخی از این خدمات هزینه‌بر هستند. این گونه خدمات برای کسب و کارهای کوچک یا کاربران خانگی بسیار مفید هستند. اما در مورد اکثر شرکت‌های متوسط و بزرگ، این گونه خدمات را باید متناسب با محیط اطلاعاتی آن شرکت ارائه کرد. بنابراین باید با توزیع کننده لینوکس در زمینه نیازهای خود مذاکره کرد.

به کارگیری لینوکس در سازمانهای مختلف متفاوت است و نمی‌توان تمامی انواع سازمانها را در این فصل گنجاند. با این حال مواردی وجود دارد که هر سازمانی قبل از مهاجرت یا اصلاح وضعیت موجود باید در نظر بگیرد.

۲-۸- پیاده‌سازی جغرافیایی

اگر سازمان شما در کشورهای مختلف دفاتری داشته باشد، باید توزیعی از لینوکس را انتخاب کنید که بتوان آن را در مقیاس جهانی به کار گرفت. در فصل ۵ آموختیم که اکثر توزیعها جهانی هستند. این احتمال وجود دارد که کارمندان شما در این دفاتر با توزیعهای محلی لینوکس آشنا باشند و حتی آن را برای استفاده در سازمان توصیه کنند. این احتمال هم وجود دارد که آن توزیع لینوکس تنها توزیعی باشد که برای آن منطقه بومی‌سازی شده است.

شما می‌توانید یک توزیع را برای استفاده در تمامی دفترها انتخاب کنید یا دفاتر مختلف را در انتخاب توزیع مورد نظر خود آزاد گذارید. در اینجا یکی از زیباییهای استاندارد LSB نمایان می‌شود. اگر برنامه‌های کاربردی شما با استاندارد LSB سازگار باشند، در تمامی توزیعهای لینوکس در هر جای دنیا می‌توان از آنها استفاده کرد.

۳-۸- مهاجرت و هم‌زیستی

همان‌گونه که در ابتدای فصل اشاره شد، ما به موارد خاص اشاره نخواهیم کرد، بلکه با نگاهی علمی به این موضوع خواهیم پرداخت. منابع زیادی در این زمینه در اینترنت موجود است که برخی از آنها در بخش مراجع این کتاب آورده شده است. می‌توان یک کتاب را به جزئیات مسائل مطرح در مهاجرت از بسترها سخت‌افزاری گرفته تا برنامه‌های کاربردی، اختصاص داد. اما در اینجا مسائل مهم مطرح در مهاجرت در لایه سامانه‌عامل را بررسی خواهیم کرد.

مهاجرت قطعاً هزینه‌بر است. برای اینکه این هزینه‌ها قابل توجیه باشند باید مزایا و صرفه‌جوئی‌های ناشی از مهاجرت را نیز در نظر گرفت. به همین دلیل است که اگر سامانه کنونی شما قابل اطمینان باشد و نیازهای تجاری شما را با حداقل هزینه برآورده کند، مهاجرت توجیه‌پذیر نخواهد بود. در ادامه برخی از هزینه‌های عمدۀ مهاجرت را بر می‌شماریم:

- سخت‌افزار- مهاجرت به لینوکس ممکن است نیازمند تغییر بستر سخت‌افزاری باشد.

بسته به اینکه چقدر از عمر سخت‌افزار موجود گذشته باشد، این گونه تغییرات ممکن است منطقی باشد. از طرف دیگر همان‌گونه که در بحثهای بعدی خواهید دید قابلیت لینوکس در پشتیبانی از بسترها سخت‌افزاری گوناگون باعث کاهش چشم‌گیر هزینه‌های سخت‌افزار می‌شود.

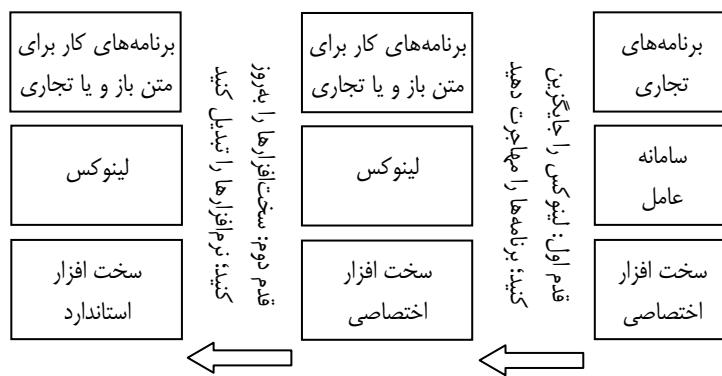
- آموزش- اگر سامانه کنونی شما عمدتاً بر اساس یونیکس بنا شده باشد، نیاز شما به آموزش کارمندان به حداقل خواهد رسید. اما اگر از ویندوز، ناول یا محیط دیگری به لینوکس مهاجرت می‌کنید، این هزینه‌ها قابل توجه خواهد بود.

• مجوزدهی- بسته به اینکه از چه نوع برنامه کاربردی در سازمان خود استفاده می‌کنید، ممکن است نیازمند ارتقاء یا خرید مجدد برخی از این گونه برنامه‌ها باشید، البته با این فرض که تمامی نرم‌افزارهای مورد نیاز شما تحت لینوکس موجود باشد. از طرف دیگر از آنجایی که هزینه تهیه اغلب توزیعهای لینوکس رایگان است، می‌توان به بودجه تهیه برنامه‌های کاربردی افزود.

- اطلاعات- دارایی واقعی سازمانها اطلاعات آنهاست، نه سختافزار و نرمافزاری که برای ذخیره و بازیابی آن اطلاعات به کار گرفته می‌شود. در انتقال از یک محیط به محیط دیگر، بیشترین هزینه و خطر مربوط به مهاجرت دادن اطلاعات سازمان می‌باشد. در این زمینه در بخش‌های بعدی بحث خواهد شد.
- برای اینکه درک بهتری از تأثیرات مهاجرت داشته باشیم نیازمند بررسی جزئیات بخش‌های مختلف هزینه‌بر در این فرآیند هستیم.

۴-۸- سختافزار

برخی افراد معتقدند، از آنجایی که لینوکس می‌تواند در بسترهای مختلف سختافزاری اجرا شود، بنابراین بر روی تمامی این بسترهای نیز به خوبی جواب می‌دهد. در فصل ۲ دلایلی را آورده‌یم مبنی بر اینکه پیاده‌سازی لینوکس در بسترهایی غیر از بستر اصلی زیرساخت اطلاعاتی سازمان، ممکن است آثار سوئی داشته باشد. با این حال استفاده مجدد از یک سختافزار در سامانه‌های غیر مرکزی زیرساخت اطلاعاتی، می‌تواند مناسب باشد. فرض کنید بر روی سختافزار کنونی شما نرمافزارهای اختصاصی پیاده شده‌اند. هدف نهایی شما این است که لینوکس را بر روی سختافزارهای استاندارد پیاده‌سازی کنید. اما چنین پیاده‌سازی را نمی‌توان یک شب انجام داد. اما می‌توان قدمهایی در زمینه استفاده مجدد از سختافزارهای موجود برداشت و فرآیند مهاجرت را در طی چند ماه تکمیل کرد.



شکل ۱-۸: مهاجرت کردن به لینوکس و سختافزارهای استاندارد

شکل ۱-۸ یک تصویری ساده از مراحل انتقال از یک محیط کاملاً اختصاصی به محیطی استاندارد و باز در اختیار می‌گذارد. مسلماً حجم کار مورد نیاز برای هر یک از این مراحل می‌تواند بسیار زیاد و پیچیده باشد. در اینجا هم اگر محیط اختصاصی موجود، یونیکس باشد، تبدیل برنامه‌ها، پرونده‌های دستور و سایر اجزاء زیرساخت نرمافزاری به حداقل می‌رسد. سامانه‌های

عامل غیریونیکس دیگر مانند VMS، MVS و ... نیازمند بازنگری عمیق قبل از استفاده در لینوکس هستند.

یکی از مشکلات مرحله اول، نرمافزارهای تجاری هستند. زیرا اکثر تولیدکنندگان این گونه نرمافزارها برنامه‌های خود را فقط تحت بسترهای سختافزاری استاندارد ارائه می‌کنند و ممکن است دریافت یک نسخه از آن نرمافزار تحت یک سختافزار خاص غیرممکن باشد. در چنین شرایطی شما باید ابتدا بر انتقال برنامه‌ها و پروندهای دستوری که خودتان نوشته‌اید، تمرکز کنید و برنامه‌های تجاری را در مرحله نهایی انتقال دهید.

در مرحله دوم یعنی تغییر سخت افزار، باید شرایط محیطی را بررسی کنید. آیا محیط کار شما برق، ابزار خنک کننده، فضا و سایر وسایل مورد نیاز برای سختافزارهای جدید را دارد؟ مسائل ارتباطی را نیز باید در نظر بگیرید. در مواردی شاید لازم باشد از وسایل ورودی و خروجی موجود برای سختافزار جدید استفاده کنید. تمامی اتصالات شبکه موجود را نیز باید بررسی کنید. این نکته را نیز در نظر داشته باشید که وقتی سختافزار را عوض می‌کنید خواص مربوط به کارآیی و قابلیت اطمینان و ارتقاء سختافزار نیز تغییر می‌کند (ممکن است بهتر یا بدتر شود). لذا باید در زمینه تأثیر این عوامل بر کاربر و کسب و کارتان بررسی دقیقی انجام دهید.

۵-۸- اطلاعات

درست است اطلاعات نیاز به بررسی بسیار دقیقی دارد، بهخصوص اگر این اطلاعات میان سامانه‌های مختلف به اشتراک گذاشته شده باشد. در مهاجرت اطلاعات درگیر مسائلی چون شیوه ذخیره اطلاعات، به اشتراک‌گذاری آن بین سامانه‌های مختلف و اطلاعات برنامه‌های کاربردی خواهد بود. اولین نکته‌ای که باید در نظر گرفته شود، ترتیب بایت Endian می‌باشد.

۱-۵-۸ - Endian

جاناتان سوئیفت در کتاب «سفرهای گالیور» توضیح می‌دهد که ساکنانی از لیلیبوت که تخم مرغ را از سمت بزرگ آن می‌شکنند، Big Endian نام دارند. در مقابل به کسانی که تخم مرغ را از سمت کوچک آن می‌شکنند، Little Endian گفته می‌شود. عبارت Endian را صنعت رایانه برای تمایز بین ترتیبهای مختلف بایت در سامانه‌های مختلف، به عاریه گرفته است. ترتیب بایت Endian که در شکل ۲-۸ نمایش داده شده است، ترتیب قرار گرفتن بایتها موقع ذخیره و بازیابی اطلاعات، در یک سامانه را تعیین می‌کند. اگر اطلاعات به صورت Big Endian ذخیره شده باشند، و سامانه دیگری این اطلاعات را به صورت Little Endian بازیابی کند، اطلاعات بازیابی شده قابل استفاده نخواهد بود و باید با برنامه‌ریزی درست از این خطا جلوگیری کرد.

E	C	A	F
0 1 0 0 0 1 1 0	0 1 0 0 0 0 0 1	0 1 0 0 0 0 1 1	0 1 0 0 0 1 0 1
اطلاعات			
آدرس حافظه	اطلاعات فوق ذخیره شده Big-Endian	اطلاعات فوق ذخیره شده به صورت Little-Endian	
00	0 1 0 0 0 1 1 0	F	0 1 0 0 0 1 0 1
01	0 1 0 0 0 0 0 1	A	0 1 0 0 0 1 0 1
02	0 1 0 0 0 0 1 1	C	0 1 0 0 0 1 1 0
03	0 1 0 0 0 1 0 1	E	0 1 0 0 0 0 0 1

شکل ۲-۸: ترتیب قرارگیری بایتهاي اطلاعات در سامانه‌های Endian

هر سامانه‌عامل یا ریزپردازنداهای ممکن است ترتیب بایت مخصوص خود را داشته باشد، لینوکس در معماریهای IA-32 و IA-64 اطلاعات را به صورت Little Endian ذخیره می‌کند. اما در معماریهای PA-RISC و SPARC به صورت Big Endian عمل می‌کند. سامانه‌عامل HP-UX تحت PA-RISC و سولاریس تحت SPARC به صورت Big Endian هستند در حالی که سولاریس تحت IA-32 به صورت Little Endian عمل می‌کند. این مشکل ترتیب بایت مهمترین مشکل در فرآیند مهاجرت است که به آن به اندازه کافی توجه نمی‌شود. همان‌گونه که می‌بینید، لینوکس در زمینه ترتیب بایتها به صورت دوگانه عمل می‌کند. این دوگانگی به نوع معماری سخت‌افزاری بستگی دارد. به همین دلیل است که برنامه‌نویسان لینوکس برنامه‌ای را که به مسئله Endian توجه کافی ندارد، ناقص تلقی می‌کنند. جامعه برنامه‌نویسان لینوکس، مجموعه‌ای از ابزار را طراحی کرده است که به کمک آنها می‌توان این ضعف را در برنامه‌ها تشخیص داد. این یکی از دلایلی است بر این گفته که برنامه‌های لینوکس به راحتی در معماریهای مختلف کار می‌کند.

اکثر برنامه‌های تجاری که تحت معماریهای مختلف ارائه شده‌اند، با این مشکل دست به گریبان بوده‌اند. این بدین معناست که اطلاعات این گونه برنامه‌ها از نظر Endian مشکلی ندارد و شما می‌توانید اطلاعات ذخیره شده تحت یک سامانه اختصاصی را در یک سامانه لینوکس بازیابی کنید. اما در زمینه برنامه‌هایی که داخل سازمان طراحی شده‌اند باید بسیار دقیق باشند، زیرا موقع طراحی اغلب این برنامه‌ها، مسائل مربوط به Endian، در نظر گرفته نشده است. بنابراین موقع انتقال اطلاعات این برنامه‌ها باید دقیق باشند.

۲-۵-۸ - سامانه‌های پرونده

در فصلهای قبلی برخی از سامانه‌های پرونده موجود در لینوکس را بررسی کردیم. متداول‌ترین آنها در لینوکس ext2 و ext3 می‌باشد. اگر از سامانه‌های یونیکس مهاجرت می‌کنید، باید بدانید که در حال حاضر از چه سامانه‌پروندهای استفاده می‌شود و آیا این سامانه‌پرونده در لینوکس نیز

قابل استفاده است. اگر از سامانه‌های ویندوز مهاجرت می‌کنید یا هم‌زمان از آنها استفاده می‌کنید،
بهتر است اطلاعات به یکی از سامانه‌های پرونده لینوکس منتقل شود. خوشبختانه جامعه متن‌باز
راهاندازهایی برای تمامی سامانه‌های پرونده میکروسافت طراحی کرده است. بنابراین در زمینه
دسترسی به پرونده‌هایی که در سامانه‌های ویندوز وجود دارند، نگرانی وجود نخواهد داشت.

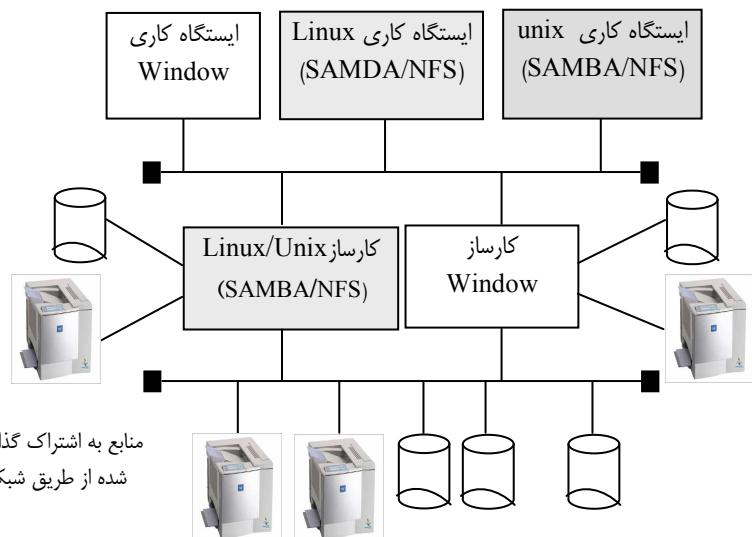
در حال حاضر سامانه‌های پرونده تجاری و باز بسیار زیادی وجود دارد که نمی‌توان آنها را در
این بحث گنجاند. جامعه متن‌باز راهانداز اکثر این سامانه‌های پرونده را طراحی کرده است.
بسیاری از شرکتهای تجاری که سامانه‌های پرونده خود را طراحی کرده‌اند، نسخه‌هایی از آن را
نیز تحت لینوکس ارائه کرده‌اند.

۳-۵-۸ - به اشتراک‌گذاری

مهمنترین نکته‌ای که بعد از طرح سامانه‌های پرونده مطرح می‌شود، سامانه‌های به اشتراک‌گذاری
پرونده‌ها و پرینترهاست. این گونه سامانه‌ها به کاربران یک شبکه اجازه می‌دهد پرونده‌ها،
پرینترها و سایر منابع خود را در شبکه به اشتراک گذارند. به کمک این سامانه‌ها کاربران
می‌توانند از منابع گران‌قیمت موجود به طور مؤثری استفاده کنند. متدالو ترین راه به
اشتراک‌گذاری پرونده‌ها و چاپگرهای استفاده از پروتکلهای میکروسافت است. این پروتکلهای
عبارتند از: CIFS^۱ و SMB^۲. سامانه‌های یونیکس به‌طور سنتی پرونده‌ها را از طریق «سامانه
پروندهای شبکه» یا NFS به اشتراک می‌گذارند. خوشبختانه جامعه لینوکس و بعضی از شرکتهای
تجاری راه حل‌هایی برای ارتباط میان این پروتکلهای بسیار مهم ارائه کردند. شکل ۳-۸ نحوه
ارتباط این خدمات را در یک شبکه نشان می‌دهد.

¹ Common Internet File System

² Server Message Block



شکل ۸-۳: مهاجرت دادن به منابع به اشتراک گذاشته شده

سامبا که از SMB گرفته شده است، یکی از بزرگترین فناوریهای متن باز است که امکان ارتباط لینوکس (و یونیکس) با محیطهای ویندوز را فراهم می‌کند. با کمک این پروتکل شما می‌توانید روند مهاجرت کارسازها و رایانه‌های رومیزی را از ویندوز با هر سرعتی که صلاح می‌دانید، ادامه دهید. سامبا هم به صورت دهنده و هم به صورت گیرنده موجود است. یک کارساز سامبا از طرف سایر رایانه‌ها همانند یک کارساز ویندوز دیده می‌شود که منابعی را به اشتراک گذاشته است.

اگر زیرساخت شما وابستگی زیادی به NFS داشته باشد این پروتکل را نیز می‌توان هم تحت لینوکس و هم ویندوز پیاده‌سازی کرد. انواع مختلف قراردادهای NFS توسط شرکتهای ثالث برای ویندوز ارائه شده‌اند.

ترکیب سامبا و NFS این امکان را در اختیار شما قرار می‌دهد که به راحتی بین ویندوز، لینوکس، یونیکس و هر سامانه‌عامل دیگری که این پروتکل‌ها را می‌شناسد، ارتباط برقرار کند. بدین ترتیب شما می‌توانید به ترکیبی ایده‌آل و با صرفه از سامانه‌های عامل مختلف مهاجرت کنید.

۸-۶- مدل‌های برنامه نویسی

منظور از مدل‌های برنامه نویسی مجموعه‌ای از توابع (API) است که برنامه نویسان از آنها برای نوشتن برنامه خود و اجرای آن تحت یک سامانه‌عامل خاص استفاده می‌کند. از آنجایی که

یونیکس و لینوکس ریشه مشترکی دارند و مدل‌های برنامه‌نویسی آنها بسیار مشابه است، برنامه‌نویسان یونیکس به راحتی می‌توانند خود را با مدل برنامه‌نویسی لینوکس وفق دهند.

مدل برنامه‌نویسی ویندوز بر اساس هسته‌ای از API‌ها به نام Win32 استوار است. این مدل با مدل‌های استفاده شده در لینوکس و یونیکس بسیار متفاوت است. به همین دلیل برنامه‌نویسان برای انتقال به لینوکس باید با یک سری مسائل دست و پنجه نرم کنند. اگر تسلط کافی بر مدل برنامه‌نویسی Win32 داشته باشید، درک مدل برنامه‌نویسی لینوکس برای شما می‌تواند بسیار مشکل باشد. آنکه برنامه‌نویسان غیریونیکس نیز با همین مشکل روبرو هستند.

اگر دانش برنامه‌نویسی سازمان بر اساس زیانهای برنامه‌نویسی سطح بالا مانند کارسازهای جوا و برنامه‌های بانک اطلاعاتی بنا شده باشد، در این صورت انتقال به یک سامانه‌عامل جدید بسیار راحت‌تر خواهد بود و حتی ممکن است از دید برنامه‌نویسان کاملاً شفاف باشد.

۷-۸- برنامه‌های کاربردی

وقتی که از یک محیط اختصاصی به یک بستر لینوکسی مهاجرت می‌کنید، باید معادلهای برنامه‌های کاربردی کنونی خود را در لینوکس بیایید. این سخن بدین معنا نیست که تمامی برنامه‌های متن‌باز می‌تواند جایگزین برنامه‌های تجاری شود، اما این نکته را نیز باید در نظر داشت که در بسیاری از موارد ما یک برنامه تجاری پرقدرت را نصب می‌کنیم، ولی از حداقل امکانات آن استفاده می‌کنیم. اگر شما از تمامی امکانات برنامه‌ای که نصب کرده‌اید، استفاده نمی‌کنید، به احتمال زیاد برنامه متن‌بازی برای شما وجود خواهد داشت که به نحو بهتری نیازهای شما را برآورده کند. به عنوان مثال اگر از یک بانک اطلاعاتی تجاری برای ذخیره و بازیابی معمولی اطلاعات کارمندان استفاده می‌کنید، پایگاه‌های داده متن‌بازی همچون MySQL و Postgres این نیاز شما را بهتر برآورده می‌کند.

۸-۸- محیط‌های رومیزی

مهاجرت یک محیط رومیزی به لینوکس باید با دقت زیادی صورت پذیرد. بسیاری معتقدند که محیط رومیزی لینوکس در حال حاضر برای کاربران حرفه‌ای یا مهندسین وارد به محیط‌های رومیزی یونیکس، مناسب است. گرچه این باور با گذر زمان تغییر خواهد کرد، ارتقاء محیط‌های رومیزی برای یک کاربر معمولی نیازمند آموزش گسترشده، در زمینه امکانات خود سامانه‌عامل و مجموعه‌های اداری می‌باشد. اگر قصد دارید تمامی محیط‌های رومیزی سازمان را به لینوکس تبدیل کنید، اول باید مجموعه برنامه‌های مورد نیاز کاربران را تهیه کنید. همچنین از اجرای درست برنامه‌های تحت وب، در مرورگرهای لینوکس اطمینان حاصل کنید.

راه حل‌های دیگری نیز امروزه موجود است که بر اساس پروژه باز WINE (شبیه‌ساز ویندوز) طراحی شده‌اند. Win32 هسته API‌های سامانه‌های عامل ویندوز است. WINE با این هدف طراحی شده است که این API‌ها را، در سامانه‌عامل لینوکس، در اختیار برنامه‌ها قرار دهد. WINE اجازه می‌دهد که تعداد محدودی از برنامه‌های کاربردی تحت ویندوز را بتوان تحت

لینوکس اجرا کرد. شرکتهای تجاری در حال آزمایش و تأیید اعتبار برنامه‌های خود تحت WINE هستند و برخی نیز ابزارهایی مخصوص نصب تحت WINE را ارائه کرده‌اند.

۸-۹- تهیه مجوز و خرید برنامه‌ها

مجوز نرم‌افزارهای تجاری برای اجرا تحت لینوکس اغلب بلاتفعیر می‌ماند. برخی از تولیدکنندگان و فروشنندگان نرم‌افزار تلاش می‌کنند ارتباطی بین هزینه بستر سخت‌افزاری و سامانه‌عامل آن با هزینه برنامه کاربردی اجرا شده در آن سامانه‌عامل برقرار کند. اما در اکثر موارد این ارتباط، از منطق درستی برخوردار نیست. در حقیقت نسبت قیمت به کارایی سخت‌افزارها در طول سالهای اخیر بسیار کاهش یافته است. در حالی که هزینه نرم‌افزارهای تجاری نه تنها کاهش نیافته، بلکه در مواردی بیشتر هم شده است. قیمت نرم‌افزارهای تجاری بر اساس رقابت موجود و امکاناتی که در اختیار مشتری قرار می‌دهد تعیین می‌شود و اغلب ربطی به سخت‌افزاری که روی آن اجرا می‌شود، ندارد.

اگر قصد دارید از یک سامانه‌عامل اختصاصی به لینوکس مهاجرت کنید، باید با فروشنندگان نرم‌افزارهای تجاری خود تماس بگیرید و از موجود بودن آن نرم‌افزارها تحت لینوکس و سازگاری آنها با LSB، اطمینان حاصل کنید.

سپس در زمینه تبدیل مجوز نرم‌افزارهای کنونی خود به لینوکس و هزینه آن مذاکره کنید. همان‌گونه که در قسمت قبل اشاره شد، در مواردی که بتوانید جایگزین متن‌بازی برای نرم‌افزار کنونی خود بیایید، هزینه خرید نرم‌افزار جدید کاهش خواهد یافت.

اگر قصد دارید برنامه‌های کاربردی متن‌باز را پیاده‌سازی کنید، باید مطمئن شوید که کارمندان از امکان کپی آزادانه این گونه نرم‌افزارها، داخل یا خارج از سازمان، آگاهند. مأمورین خرید شما باید فرق بین نرم‌افزار متن‌باز و اختصاصی مشخص شد، مسلماً مایل خواهید بود روند تهیه نرم‌افزارهای متن‌باز را نیز تعیین کنید. فروشنندگان نرم‌افزارهای متن‌باز باید شما را از قیمت تک‌تک اقلام خریداری شده همراه نرم‌افزار، مانند رسانه (لوح فشرده حاوی برنامه)، دفترچه‌های راهنمای پشتیبانی و غیره مطلع نماید. در مواردی ممکن است فروشنندگان سخت‌افزار مجموعه‌های لینوکس یا نرم‌افزارهای متن‌باز را در سخت‌افزار خود قرار دهد و بابت این کار از شما هزینه دریافت کند. شما باید از ریز این هزینه‌ها مطلع باشید و در زمینه کارا بودن نرم‌افزار در سازمان تصمیم‌گیری کنید. باید بر روی آموزش مأمورین خرید خود در زمینه نحوه کار مدل متن‌باز وقت کافی بگذارید. این در نهایت به نفع شما خواهد بود.

۱۰-۸- پشتیبانی

نحوه پشتیبانی در لینوکس و متن‌باز ممکن است با سامانه فعلی شما تفاوت بسیاری داشته باشد. خوشبختانه در این زمینه حق انتخاب تماماً با شماست. روش‌های مختلف پشتیبانی را می‌توان به سه دسته تقسیم‌بندی کرد:

- هر نرم‌افزار یک شرکت

- تمامی نرم افزارها، یک شرکت
- استفاده از پشتیبانی جامعه لینوکس و متن باز از آنجایی که این روش‌های پشتیبانی احتمالاً با روش‌های موجود شما بسیار فرق می‌کند، هر یک از این روشها را با جزئیات بیشتری بررسی می‌کنیم.

۸-۱-۱- هر نرم افزار یک شرکت

در این روش برای هر جزئی از مجموعه نرم افزارهای استفاده شده با یک شرکت پشتیبانی کننده قرارداد بسته می‌شود. معمولاً پشتیبانی سخت افزار را فروشنده آن و پشتیبانی سامانه عامل را توزیع کننده آن ارائه می‌دهد. پشتیبانی هر یک از نرم افزارها را نیز ممکن است فروشنده آنها ارائه دهد.

بزرگترین مزیت این روش این است که اگر سازمان از عملکرد هر یک از شرکتها ناراضی باشد، می‌تواند آن نرم افزار یا شرکت پشتیبانی کننده را عوض کند، بدون اینکه تأثیری بر پشتیبانی سایر نرم افزارها بگذارد. مزیت دیگر این روش این است که می‌توان برای هر یک از برنامه‌ها بهترین پشتیبانی را تأمین کرد.

این روش پشتیبانی برای محصولات متن باز تجاری و اسم و رسم‌دار مناسب‌تر به نظر می‌رسد، مانند توزیعهای لینوکس. به عنوان مثال اگر شما توزیع لینوکس خود را از یکی از فروشنده‌گان معتبر دریافت کنید، آن فروشنده در بهترین موقعیت برای رفع مشکلات شما و اصلاح توزیع مناسب با نیازهای شما می‌باشد. حتی در برخی از موارد شما می‌توانید از رقابت ایجاد شده در ارائه خدمات یک محصول سود ببرید. آنچه که گفته شد، می‌توان به تمامی محصولات متن باز تعمیم داد.

با وجود اینکه بسیاری از شرکتهای استفاده کننده از لینوکس و متن باز این مدل را ترجیح داده‌اند، اما معایبی نیز دارد:

- تداخل بین مجموعه برنامه‌های مورد استفاده - همان گونه که گفته شد استفاده از این روش می‌تواند بهترین خدمات را به ارمغان آورد. با این حال در مواردی شما به مشکلاتی برمی‌خورید که منبع آنها را نمی‌دانید. در نهایت مجبور می‌شوید تداخلهای موجود بین پشتیبانی کننده‌ها را بررسی و برطرف کنید.
- ارتقاء نرم افزارها - مدیریت و نظارت بر عملکرد فروشنده‌ها و هر یک از قراردادهای پشتیبانی بدین معناست که شما مالکیت نرم افزارهایتان را در اختیار خود دارید. اگر یکی از این نرم افزارها را ارتقاء دهید، ممکن است شرایطی بوجود آید باعث لغو قرارداد پشتیبانی دیگر شود. بنابراین در این روش باید شرایط پشتیبانی هر یک از فروشنده‌گان در قرارداد آنها به خوبی مشخص باشد و موارد احتمالی تداخل ایجاد شده بین نرم افزارها و نحوه برطرف کردن آن در نظر گرفته شده باشد.

• مدیریت قراردادها- در این روش نظارت دقیق بر هر یک از قراردادهای پشتیبانی الزامیست. سهلانگاری در این زمینه، به شکست این روش پشتیبانی در سازمان، منجر خواهد شد.

بنابراین معایب این روش را همانند مزایای آن به خوبی در نظر بگیرید و با دید بهتری این روش را انتخاب یا رد کنید.

۲-۱۰-۸- تمامی نرم افزارها، یک شرکت

این روش نقطه مقابل «هر نرم افزار، یک شرکت» قطب مخالف هر نرم افزار یک شرکت می باشد. در این روش، از پشتیبانی یک شرکت برای تمامی نرم افزارهای خود بهره می برید. مزیت این روش واضح است: شما با یک شرکت طرف حساب خواهید بود. در این روش شرکت مورد نظر، مجموعی از خدمات را باید در اختیار شما قرار دهد. از آنجایی که محصولات پشتیبانی شده متن باز هستند، این شرکت می تواند نقش اصلاح کد آنها را نیز در موارد برخورد با مشکل، بر عهده گیرد. در روش «تمامی نرم افزارها، یک شرکت» این امکان وجود دارد که شرکت طرف حساب شما، پیمانکار اولیه باشد و پشتیبانی هر یک از اجزاء را به پیمانکارهای فرعی واگذار کند. صرف نظر از تصمیم پیمانکار وی باید از نقاط تأثیرگذار در جامعه متن باز باشد و صلاحیت اصلاح کد برنامه ها را احراز کرده باشد.

۲-۱۰-۳- استفاده از پشتیبانی جامعه لینوکس و متن باز

سازمانهای فناوری اطلاعات بزرگ، ممکن است به این نتیجه برسند که خودشان محصولات استفاده شده را پشتیبانی کنند و در موارد خاص از شرکتهای تجاری بهره گیرند و بدین ترتیب از هزینه های پشتیبانی بکاهند. در این روش یک تیم فنی خبره و دانشآموخته مورد نیاز است. برای ایجاد چنین تیمی، باید آنها را در برقراری ارتباط با بازیگران اصلی جامعه متن باز آزاد گذاشت. هدف شما باید این باشد که نیازهای پشتیبانی خود را برآورده کنید و در عین حال می توانید به پیشرفت جامعه متن باز نیز کمک کنید. در این زمینه نباید از ابتدا انتظار داشته باشید همانقدر که به جامعه متن باز خدمت می کنید، از آن بهره برداری کنید، بلکه با کمی صبر ثمره آن را خواهید دید. در این روش، بین دو طرف (سازمان و جامعه متن باز) وابستگی ایجاد می شود و از آن سود می برند.

اگر پشتیبانی داخلی با پشتیبانی تجاری همراه شود، در این صورت نتیجه این خواهد شد که کارمندان شما نقش اصلی را در یافتن مشکلات و برطرف کردن آنها ایفا خواهند کرد. پس از یافتن مشکل، اغلب باید کد منبع اصلاح شود و اصلاحات برای نگهدارنده اصلی کد منبع فرستاده ارسال شود. سپس در صورتی که از خدمات تجاری نیز استفاده می کنید، آنها را از اصلاحات انجام شده در برنامه مطلع سازید تا در نسخه های بعدی از آنها بهره گیرند. همچنین در قرارداد خدمات پشتیبانی تجاری باید شرکت پشتیبانی کننده ملزم به اعمال کردن اصلاحات برنامه تا زمان عرضه نسخه بعدی باشند.

مزیت اصلی این روش پشتیبانی این است که سازمان شما نقش اصلی را ایفا خواهد کرد و نظارت مستقیمی بر مشکلات ایجاد شده و علت آنها خواهد داشت. در روشهای پشتیبانی تماماً تجاری، شما باید امیدوار باشید که آن شرکت، حیاتی بودن آن مشکل را درک کند. در غیر این صورت باید خرر ناشی از تأخیر ایجاد شده در رفع مشکل را تقبل کنید. اما در مدل ذکر شده در بالا، تصمیم‌گیر اصلی خودتان هستید، خودتان تصمیم می‌گیرد که آیا برای تغییرات یا اصلاحات سرمایه‌گذاری شود یا نه و در این صورت خودتان مستقیماً با تولیدکننده آن نرمافزار در راستای اعمال آن تغییرات کار خواهید کرد. مادامی که تغییرات یا اصلاحات درخواستی شما در راستای حرکت آن پروژه باشد (پروژه‌ای که نتیجه آن، نرمافزار مورد استفاده شما بوده است)، مدیران پروژه از اعمال تغییرات شما خوشحال نیز خواهند شد، البته با این فرض که سابقه خوبی در خدمت به جامعه متن‌باز داشته باشید.

۴-۱۰-۸ - تأثیرگذاری و ارتباطات

صرفنظر از روش پشتیبانی که شما انتخاب می‌کنید، رمز موقعیت شما در «تأثیرگذاری و ارتباطات» است. سؤال اولی که در این راستا باید پاسخ داده شود این است که چه کسی قرار است رابطه مایبن سازمان شما و نگهدارندگان کلیدی [کدهای منبع] برقرار کند. شما می‌توانید این وظیفه را به فروشنده‌گان نرم‌افزارها محول کنید یا خودتان آن را بر عهده گیرید. راه سوم هم این است که از ترکیبی از این دو روش استفاده کنید. این یکی از مزایای اصلی نرم‌افزارهای متن‌باز است که شما می‌توانید کنترل و نظارت محیط را خودتان بر عهده گیرید و در انتهای از ساختار خدمات‌رسانی رقابتی موجود نیز بهره گیرید. ممکن است برخی از شرکتها چنان به خدمات تجاری وابسته شده باشند که نتوانند روش خود را عوض کنند تا چه رسدد بخواهند از روشی استفاده کنند که نیازمند رابطه مستقیم با جامعه متن‌باز و تأثیرگذاری بر آن باشد. اگر قصد دارید از مزایای متن‌باز بهره گیرید ولی با روش پشتیبانی از طریق تأثیرگذاری مشکل دارید، در این صورت بهترین راه این است که با شرکتهایی تجاری کار کنید که سابقه خوبی در ارتباط با جامعه متن‌باز دارند.

۱۱-۸ - آموزش

اکثر فروشنده‌گان سامانه‌عامل برنامه‌هایی برای مدرک دادن به کاربران دارند. به کمک این مدرکها، شرکتها می‌توانند از سطح اشتغالی کارمندانشان با برنامه موردنظر آگاهی یابند. امروزه گذاندن آموزش‌های استاندارد، یک برگ برنده تلقی می‌گردد. این آموزشها برای فروشنده سامانه‌عامل نیز مزایایی دارد از جمله اینکه متخصصان فناوری اطلاعات در انجام هرچه بهتر نصب و پیکربندی آن سامانه‌عامل با یکدیگر رقابت می‌کنند و وجهه بهتری از آن شرکت به نمایش می‌گذارند. مزیت آن برای فروشگاه‌های محصولات فناوری اطلاعات نیز این است که معیار بهتری برای استخدام کارمندان و ارزیابی دانش موجود در زمینه پیاده‌سازی و پشتیبانی نرم‌افزارها، در اختیارشان قرار می‌دهد. برای سازمانها نیز این آموزشها، رزومه تحصیلی کارمندان آنها را سنگین‌تر می‌کند که در نهایت به سطح دانش آن سازمان می‌افزاید.

نیاز به استانداردسازی آموزش‌های لینوکس مستقل از توزیع کننده، منجر به تأسیس مؤسسه حرفه‌ایلینوکس یا¹ LPI (www.lpi.org) شد. LPI سه مدرک در سه سطح ارائه می‌کند. سطح اول برای کاربران حرفه‌ای و پشتیبانی معمولی لینوکس مناسب است. سطح دوم برای طرح ریزی و پیاده‌سازی شبکه‌های کوچک طراحی شده است و سطح سوم نیز برای معماران ارشد فناوری اطلاعات که وظیفه طراحی و پیاده‌سازی زیرساخت شرکتهای بزرگ را بر عهده دارند، در نظر گرفته شده‌اند.

برنامه‌های آموزشی دیگری نیز برای لینوکس از جمله Comptia (www.comptia.org) و SAIR (www.linuxcertification.org) وجود دارد. همچنین برخی از توزیع کننده‌گان لینوکس نیز مدرکهایی برای توزیع خود ارائه می‌کنند. سازمان شما باید جزئیات هر یک از این مدرکها را مطالعه کند و در زمینه پذیرش یا عدم پذیرش هر یک از آنها تصمیم گیرد. این کار، در تهیه برنامه آموزشی کارمندان موجود و در حال استخدام، به شما کمک می‌کند.

در اغلب این برنامه‌های آموزشی، کار با برخی از اجزاء موجود در اکثر لینوکس‌ها، مانند سامبا و آپاچی، آموزش داده می‌شود. اما در هیچ یک از آنها اشاره‌ای به نرمافزارهای کاربردی مورد استفاده شما نمی‌شود. بنابراین باید پس از اینکه این نرمافزارها را تعیین کردید به دنبال دوره‌های آموزشی آنها برای کارمندان خود باشید. از طرف دیگر شما می‌توانید برنامه آموزش کارمندان را نیز خودتان طراحی کنید. این هم یکی دیگر از مزایای تکنولوژی متن‌باز است. از آنجایی که کد منبع همه پروژه‌ها در دسترس است، شما می‌توانید برنامه‌های آموزشی را تا هر سطحی که صلاح می‌دانید؛ طرح ریزی کنید. حتی می‌توانید از این راه کسب درآمد کنید.

در نهایت آموزشی که باقی می‌ماند، یادگیری فلسفه متن‌باز است. این آموزشها شامل درک مجوزها و ملاحظاتی است که کارمندان موقع پیاده‌سازی و استفاده از نرمافزارهای متن‌باز باید مدنظر داشته باشند. بخش سوم این کتاب نحوه استفاده از نرمافزارهای متن‌باز در یک سازمان را بررسی می‌کند. همچنین نحوه عرضه کردن نرمافزارهای داخلی سازمان، خود تحت مجوزهای متن‌باز بررسی خواهد شد. فراموش نکنید که آموزش جزئیات این فرآیندها، مهمترین بخش برنامه آموزشی شما می‌باشد.

¹ Linux Professional Institute

۱۲-۸ - جمع‌بندی

در این فصل، زیربنای دانش شما در زمینه لینوکس و متن‌باز تکمیل شد. با مطالعه این فصل شما باید به درک عمیقتری از مراحل عملی پیاده‌سازی لینوکس رسیده باشید. مسلماً آسانترین مهاجرت به لینوکس از محیط‌های یونیکس می‌باشد. اما مهاجرت از سایر محیط‌ها نیز تاکنون موفقیت‌آمیز بوده است.

اکنون که درک بهتری از لینوکس پیدا کرده‌اید، زمان آن رسیده است که اعلام کنیم لینوکس خود یکی از پروژه‌های بزرگ متن‌باز می‌باشد. همان‌گونه که تاکنون چندین بار اشاره شد، آنچه تاکنون آموخته‌اید، نه تنها به لینوکس بلکه به تمامی پروژه‌های متن‌باز قابل تعمیم است. در بخش سوم متن‌باز را با جزئیات بیشتری بررسی خواهیم کرد و در زمینه گسترش آموخته‌های لینوکس به هر پروژه متن‌باز دیگری بحث خواهد شد.

بخش سوم نقش متن‌باز در کسب و کار

در بخش سوم نگاه دقیق‌تری به جنبه‌های مختلف تجاری متن‌باز می‌اندازیم. کلمه «متن‌باز» طیف گسترده‌ای از مفاهیم را شامل می‌شود از جمله مجوزدهی نرم‌افزارها، فرآیند و فرهنگ تولید آنها و غیره. جا انداختن مفاهیم متن‌باز در یک سازمان امری پیچیده و مشکل است که در صورت تحقق آن، سود قابل توجهی برای آن سازمان به ارمغان خواهد آورد.

در فصل اول این بخش، فرآیند تولید نرم‌افزار متن‌باز در یک سازمان بررسی خواهد شد. این فصل درک بهتری از فرهنگ و روش‌های تولید برنامه متن‌باز در اختیار شما قرار می‌دهد. فصل دوم این بخش در زمینه تأثیر اقتصادی نرم‌افزارهای متن‌باز بر ساختارهای سنتی کسب و کار بحث می‌کند. در فصل بعدی نگاه عمیق‌تری به مدل‌های اقتصادی متن‌باز خواهیم انداخت. در فصل دوازدهم، استفاده از مجوز متن‌باز برای نرم‌افزارهای موجود و به کارگیری نرم‌افزارهای متن‌باز در تجارت بررسی خواهد شد. آخرین نکته در زمینه متن‌باز، نحوه سنجش منابع انسانی و تأثیر آن بر به خدمت گرفتن بهترین استعدادها می‌باشد.

فصل نهم بازار شرکتی

من همیشه گفته‌ام: «مسئله اصلی لینوکس نیست، بلکه فلسفه متن باز است» یا به عبارت دقیق‌تر فرآیند طراحی نرم‌افزارهای متن‌باز، اریک ریموند، به روش طراحی و تولید نرم‌افزارهای متن‌باز، «بازار» اطلاق کرده است که رمز پیشرفت لینوکس می‌باشد. در این فصل ما روش طراحی بازار را به شرکتها تعیین می‌دهیم. در این فصل شما مفاهیم زیر را خواهید آموخت:

- تفاوت بین روشهای طراحی «کاتدرال» و «بازار»
- ساختار بازار در یک شرکت از نظر تئوری
- چگونگی بهره‌گیری از روش طراحی بازار در شرکت

یکی از نکات کلیدی که هنگام خواندن این فصل باید مدنظر داشته باشید این است که لزومی ندارد اطلاعات پروژه‌های خود را در اختیار عموم قرار دهیم. در این فصل فقط روشهای توسعه متن‌باز مورد بحث قرار می‌گیرد. توجه داشته باشید که استفاده از روشهای توسعه متن‌باز به این معنا نیست که باید برنامه‌های خود را تحت مجوزهای آن عرضه کنید. در فصل دوازده به طور کامل نحوه عرضه کد پروژه‌های انجام شده در داخل سازمان را مورد بحث قرار خواهیم داد. بنابراین اگر شما پروژه‌ای را در داخل سازمان با استفاده از روشهای توسعه متن‌باز پیش می‌برید، مجبور نیستید که اطلاعات آن پروژه را در اختیار عموم قرار داده و تحت مجوزهای متن‌باز منتشر کنید.

تشریح راهکارهای توسعه متن‌باز برای یک مدیر دوره‌ای ممکن است بسیار مشکل باشد، بنابراین در این فصل ابتدا ساختار فرضی یک سازمان را تعیین می‌کنیم. این کار دو مزیت دارد: ۱) روشهای طراحی به عباراتی ساده‌تر و آشناتر معرفی می‌شوند. ۲) شما را مجبور به تأمل در زمینه اجزاء و معماری نرم‌افزار موجود در سازمان می‌کند.

۹-۱- کاتدرال و بازار

این مفاهیم از مقاله معروف اریک ریموند گرفته شده‌اند. در این مقاله او روش طراحی سنتی نرم‌افزار را که به خوبی از آن آگاه هستیم، به طراحی و ساخت یک کاتدرال (کلیسای بزرگ) تشبیه می‌کند. وی در مقابل روشهای نوین طراحی متن‌باز را که لینوس توروالدز از پیشروان آن است، به روش گسترش بازار تشبیه می‌کند. من قصد ندارم که تمامی مطالب مقاله ریموند را بررسی کنم اما لازم است که در ک درستی از روشهای توسعه کاتدرال و بازار داشته باشیم.

- روش توسعه کاتدرال - در این روش سنتی، سعی می‌شود که تیمهای کاری کوچک نگه داشته شوند. نحوه طراحی و عملکرد قبل از آغاز ساخت، باید به خوبی مشخص شده باشند. نقشه کامل عملیات از معماری، طراحی تا اجرا و آزمایش به طور دقیق تعیین و مستند شده است. مادامی که طراحی نرم‌افزار کامل نشده است، به بازار عرضه نمی‌شود و نظرات کاربران اکثرًا از طریق نسخه‌های آزمایشی عرضه شده آلفا و بتا دریافت می‌شود.

- روش توسعه بازار - نگهدارنده یک نرمافزار وقتی آن را عرضه می‌کند که حداقل کارابی را داشته باشد و دیگران بتوانند آن را تکمیل کنند، این بدان معناست که نرمافزار عرضه شده ناقص بوده و حتی ممکن است ایراداتی نیز داشته باشد. چرخه‌های عرضه نرمافزار بسیار زیاد و با فاصله‌های زمانی اندک (حتی یک ساعت) صورت می‌گیرند. بازخوردها بالفاصله بررسی می‌شوند. هر کسی در هر جای دنیا که ایده خوبی برای بهبود برنامه دارد می‌تواند خودش این ایده را به برنامه اضافه کند یا در رفع اشکالات آن تلاش کند. زمان عرضه جدی محصول را نگهدارنده نرمافزار به صلاحیت خود تعیین می‌کند.

آنچه که اشاره شد توضیح بسیار مختصری از این دو مفهوم بود. بنابراین توصیه می‌کنم مقاله (یا کتاب ریموند) را خودتان مطالعه کنید. ریموند قصد داشت موقفيت لینوس توروالدز را در زمینه لینوکس تقلید و تکرار کند. او توانست روش توروالدز را که کار کردن با جامعه‌ای از داوطلبان بود، در پروژه دیگری اعمال کند. آنچه که ریموند به آن اشاره نکرده است این است که چگونه می‌توان از مزایای روش توسعه بازار بهره برد و آنها را در دنیای تجارت به کار گرفت. سؤال اصلی اینجاست که چگونه می‌توان ساختار واقعی شرکتهای بزرگ امروزی را در برنامه‌ریزیها در نظر گرفت. از جمله بخش‌های ساختاری می‌توان به این موارد اشاره کرد: سلسه مراتب مدیریتی، اهداف شغلی کارمندان، قوانین منابع انسانی، در دسترس بودن تجهیزات، سیاستهای راهبردی که توسط هیئت مدیره تعیین شده باشد و فشارهای واردہ از طرف شرکتهای رقیب.

۲-۹- ساختار، سیاستها را دنبال می‌کند

قبل از اینکه به سراغ یک مدل عملی برویم، باید اهداف و سیاستها را مشخص کنیم. اشتباه بسیاری از مدیران این است که قبل از تعیین سیاستهای شرکت، ساختار سازمانی را وضع می‌کنند. مسلماً نمی‌توان در این مجال، اهداف و سیاستهای همه شرکتها را در نظر گرفت، اما اهداف مشترکی برای اکثر شرکتها وجود دارد که قابل بحث است. اگر قصد دارید روش توسعه بازار را در شرکت خود پیاده کنید، باید بر تجربه‌های جامعه متن‌باز در استفاده از این روش، مروری داشته باشید:

- توسعه سریع - اکثر صاحب نظران بر این باور هستند که جامعه متن‌باز از چرخه‌های تولید قابل توجهی برخوردار است. لینوکس در مدت ۱۰ سال تکامل یافت و تبدیل به یک سامانه‌عامل پیشرفته و قابل اطمینان شد. دو رابط کاربری گرافیکی به GNOME و KDE در مدت کمتر از ۲ سال برای آن طراحی شد. البته باید به موقفيت وصف‌نایزی پراستفاده‌ترین برنامه متن‌باز، کارساز وب آپاچی، نیز اشاره کرد.
- توسعه توزیع شده - تنها دلیل توسعه صحیح لینوکس و آپاچی را باید توسعه سریع و همزمان اینترنت دانست. جامعه متن‌باز چنان به خوبی از این زیرساخت ارتباطی بهره گرفت که نرمافزارها تولید می‌شدند بدون اینکه نویسنده‌گان آنها یکدیگر را دیده باشند.

- این توانایی متن باز در به کارگیری توسعه توزیع شده، آن را قادر می‌ساخت که به بهترین استعدادها صرفنظر از اینکه کجای جهان باشند، دسترسی یابد.
- بهترین استعدادها- از مشخصات جامعه متن باز این است که کارهای ضعیف را قبول (و حتی تحمل) نمی‌کند. اگر اهل انجام دادن کار اصولی (کد منبع، طراحی، معماری، آزمایش، مستندسازی و ...) نیستید، نمی‌توانید به جامعه متن باز راه یابید. البته من تا به حال به عضوی از این جامعه برنخوردهام که چنین دیدی داشته باشد، اما یک نوع مدیریت استعداد به طور ذاتی در این جامعه وجود دارد. مدیریت استعدادها از مشکل‌ترین جنبه‌های مدیریت در یک سازمان است.
 - برآوردن نیاز کاربران- یکی از مهمترین خاصیت‌های جامعه متن باز این است که برنامه‌نویس، کاربر برنامه نیز هست. بنابراین دور از انتظار نیست که برنامه‌ها متناسب با نیاز کاربران نوشته شود. این مهم باید در سازمان شما پذیرفته شود که برنامه‌نویسان، کاربران برنامه نیز هستند و نظرات آنها، سهم بسزایی در بهبود پروژه خواهد داشت.
 - کیفیت- ضرب المثل «اگر به اندازه کافی چشم وجود داشته باشد، همه اشکالات برنامه (باگها) کم عمق می‌شوند» را ریموند در مقاله ابداع کرد و امروزه، تبدیل به ضرب المثلی رایج در جامعه متن باز شده است. روش توسعه به ظاهر بدون کنترل متن باز، منجر به تولید با کیفیت‌ترین نرم‌افزارها شده است. در اینجا منظور از کیفیت این است که یک سامانه بتواند منطبق با تمامی نیازهایی که به خاطر آنها طراحی شده است، عمل کند. مدیران با تجربه پروژه‌های نرم‌افزاری ادعا می‌کنند که برای داشتن کیفیت، باید از ابتدا ساختار پروژه معین و قابل درک باشد. اگر نیاز کاربران فراتر از معماری طراحی شده برای آن سامانه باشد، در این صورت باید اکثر اجزاء سامانه مجدد طراحی و برنامه‌نویسی شود. اگر بخواهید کیفیت را از معماری به خود برنامه گسترش دهید، یک دید کلی از سامانه باید موجود باشد تا آن را به درستی هدایت کند. آنچه که در بالا آمد لیست قابل توجهی از مزایای مدل بازار است. برای قانع شدن مدیران، شاید نیازی به ذکر همه این مزیتها نباشد. اما مخالفین این روش توسعه، به برخی از کاستیهای آن اشاره می‌کنند که می‌تواند آن را از موققیت بازدارد.
 - نیازهای بازار- این خوب است که برنامه‌نویسان، کاربر برنامه نیز باشند اما اغلب استفاده کنندگان از برنامه‌ها، برنامه‌نویس نیستند. در نتیجه، محصول نهایی مطابق با نیازهای واقعی مشتریان نخواهد بود.
 - مسیر توسعه- اغلب ما، پروژه‌های توسعه را مانند راههایی می‌دانیم که ارتباطات داخل و خارج سازمان را تسهیل می‌کند. درک این ایده که برنامه‌نویسان وقتی برنامه‌شان را عرضه می‌کنند که آماده شده باشد، برای مدیریت و بخش بازاریابی شرکت مشکل است.

- مدیریت مردمی - چه کسی مدیریت این برنامه‌نویسان حرفه‌ای را بر عهده دارد؟ آنها طی چه فرآیندی هدایت می‌شوند؟ مدیریت تیمی از برنامه‌نویسان که استخدام سازمان شما نیستند، کار بسیار مشکلی است.

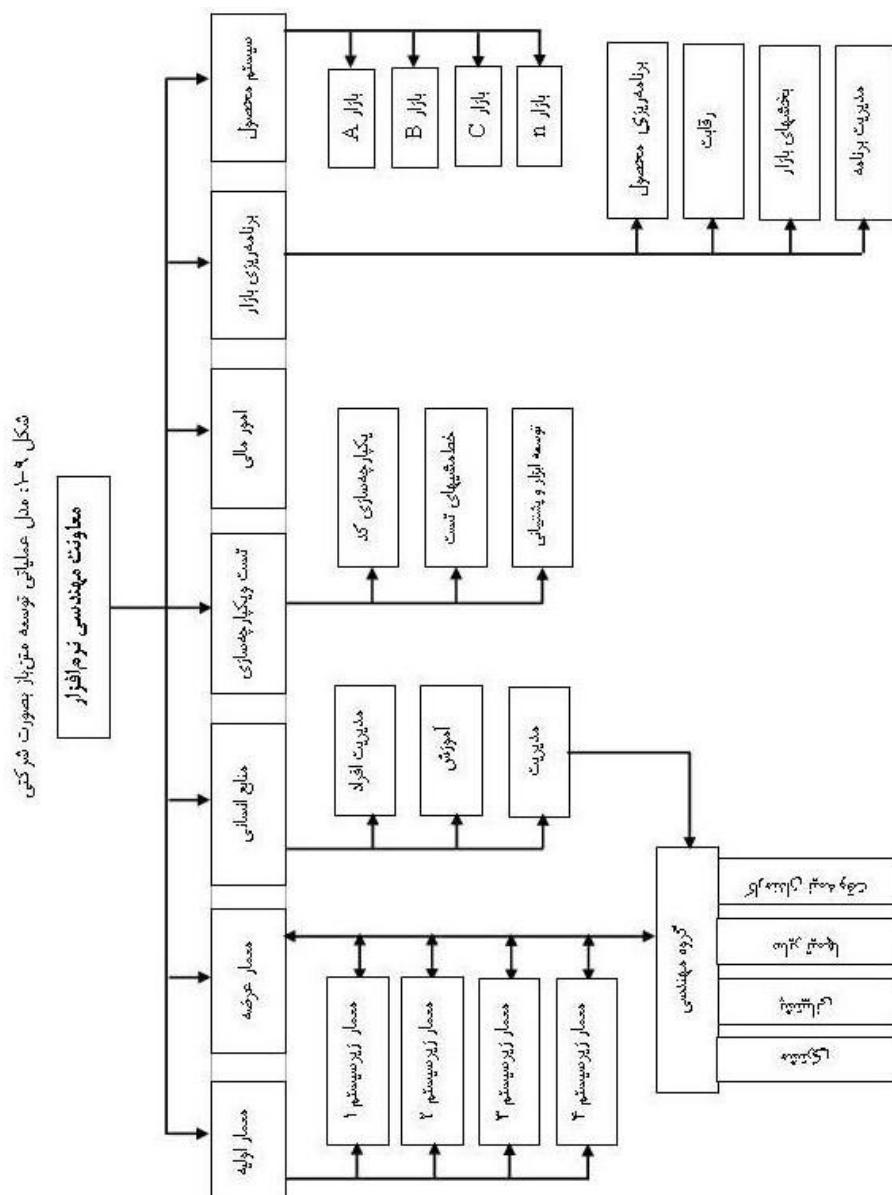
این کاستیها اجازه نمی‌دهند که مدیر کنترل دلخواه خود را بر پروژه داشته باشد و بتواند در مورد آنچه که عرضه شده است، پاسخگو باشد. آنچه که شما باید یاد بگیرید این است که روشهای کنترل سنتی را کنار بگذارید تا بتوانید محصولات بیشتری با کارایی بالاتر و کیفیتی فراتر از انتظار عرضه کنید. از مزایای طبیعی این گونه عملکرد این است که شما نیروی کاری را در اختیار خواهید گرفت که برای کسب و کار شما بهینه شده‌اند و در عین حال به کار خود عشق می‌ورزند.

۳-۹- بازار ساخت یافته

حال که به درک خوبی از اهداف مورد نظر خود رسیده‌اید و از مشکلات احتمالی نیز آگاهی دارید، من می‌توانم یک مدل عملی برایتان ارائه کنم که در شکل ۱-۹ به تصویر کشیده شده است. مدل عرضه شده یک تیم برنامه‌نویسی بزرگ (بیش از ۱۰۰ مهندس رایانه) را دربرمی‌گیرد. فرقی نمی‌کند که شما در حال تولید یک محصول کوچک تجاری هستند و یا پروژه‌ای بزرگ برای استفاده در داخل شرکت اجرا می‌کنید. فراموش نکنید که در این فصل هدف شما این نیست که کد برنامه‌تان را در اختیار عموم قرار دهید، شما فقط می‌خواهید از مزایای توسعه توزیع شده، رایج در جامعه متن‌باز، بهره ببرید.

مثال ذکر شده، تصویر جامعی از مراحل توسعه نرم‌افزار به صورت توزیع شده در اختیار شما قرار می‌دهد. شما باید این ساختار عملیاتی را در ساختار سازمانی خود منعکس و اعمال کنید. قبل از اینکه هر بخش از این ساختار را بررسی کنید، سعی کنید بخشهایی از ساختار سازمان خود را که با این ساختار متفاوت است، پیدا کنید. به عنوان مثال یکی از نکات اختلاف با سازمان شما ممکن است این باشد که در این ساختار تمامی دانش مهندسان در یک قسمت مرکز شده است. همچنین دقت کنید که اکثر فرآیند توسعه پرsonل به جای اینکه در زنجیره مدیریت صورت گیرد، در منابع انسانی انجام می‌شود. این موارد به زودی توجیه خواهند شد. در حالی که هر بخش از این ساختار را بررسی می‌کنید، در نظر داشته باشید که وظیفه شما این است که یک سامانه‌عامل لینوکس قابل استفاده را پیاده‌سازی کنید (مثلاً به صورت پرتال، بانک اطلاعاتی و یا محصولی خاص). اکنون شما باید بتوانید این دو نکته را به یکدیگر مرتبط کنید که جامعه متن‌باز چگونه پروژه‌های متن‌باز را اجرا می‌کند و شما چگونه از روشهای آنها برای پروژه‌های خود استفاده می‌کنید. با فرض اینکه شما در حال توسعه سامانه‌عامل لینوکس هستید، می‌توانید تناظر یک به یکی بین جامعه متن‌باز و سازمان خود برقرار کنید و نکته اصلی همینجاست. مسلماً تفاوت‌هایی نیز وجود خواهد داشت اما نگذارید این تفاوت‌ها شما را از به کارگیری این مزایا دور کند. در این روند، شما دید خوبی از نحوه توسعه نرم‌افزارها در این جامعه بدست خواهید آورد.

این ساختار، درگیر مسائل مربوط به فروش، تولید و پشتیبانی نخواهد شد. در انتهای این فصل تأثیر آن را بر این گونه مسائل بررسی خواهیم کرد.



۴-۹- معاونت مهندسی نرم افزار

فکر خود را مشغول پیچیدگیهای ساختار ارائه شده نکنید. در اینجا می‌خواهیم وظایف بالاترین مدیر مسئول در زمینه عرضه محصول به بازار را بررسی کنیم. وظیفه این معاونت این است که سیاستهای مربوط به محصول در حال تولید را طرح‌ریزی کرده، تولید آن را آغاز کند. مثلاً بداند که هسته لینوکس را طراحی می‌کند یا کارساز آپاچی را؟ اغلب لازم است این پروژه را با سایر پروژه‌های در حال اجرا با سازمان مرتبط کرد. معاونت مهندسی نرم افزار باید ارتباط بین پروژه‌ها را بررسی و مستندسازی کند. با توجه به اینکه سازماندهی کارمندان در ساختار سازمانی ارائه شده مشخص است، او تمامی ابزار مورد نیاز را در اختیار دارد تا بتواند تشخیص دهد که آیا پیشرفت انجام شده هم‌جهت با سیاستهای سازمان است یا خیر. سوالات اصلی که معاونت مدیریت نرم افزار باید به آنها پاسخ دهد عبارتند از:

- آیا ارتباط تیم بازاریابی با تیمهای مهندسی به طور مؤثر برقرار است؟ آیا بازار مورد نظر آنها مناسب است؟
- آیا گروه منابع انسانی توانسته است استعدادهای مورد نیاز را استخدام کند تا پروژه با موفقیت انجام شود؟
- آیا محصول در حال توسعه به درستی بر نیازهای بازار مورد نظر متمرکز شده است؟
- آیا درآمد حاصل از یک محصول منطبق با انتظارات سازمان است؟
- آیا خلاقیتها و نوآوریهای متناسب با نیاز بازار (یا سریعتر از آن) پیش می‌رود؟
- آیا تمامی اجزای مورد نیاز برای یک محصول عرضه شده‌اند؟
- آیا کیفیت محصول قابل قبول است؟ آیا سرعت واکنش ما به اشکالات محصور قابل قبول است؟
- آیا این محصول نیازهای کاربران خود را به خوبی برآورده می‌کند؟
- آیا نمودار نسبت هزینه به درآمد مطابق با انتظارات پیش می‌رود؟
- آیا دلگرمیهای سازمان منجر به بالا رفتن انرژی تیمها شده است؟
- آیا از بهترین استعدادهای موجود استفاده کرده‌ایم؟

اگر معاونت مهندسی نرم افزار این سوالات را سرلوحة خود قرار دهد، بهترین و خلاقانه‌ترین محصولات را با بالاترین کیفیت و سود تضمین شده به کاربران عرضه خواهد کرد. خوب بودن محصول خود باعث دلگرمی تیم و فعالیت بهتر آن می‌شود، زیرا اگر تیمی موفق شود، افتخار آن برای خود آنها خواهد ماند.

این معاونت وظیفه مهم دیگری نیز دارد. به زودی خواهید فهمید که معمار اولیه پروژه زمان آماده شدن محصول برای عرضه را اعلام خواهد کرد. در اکثر سازمانها فشار زیادی وارد می‌شود تا محصول زودتر عرضه شود اما در اینجا معاونت مهندسی نرم افزار باید صبر کند تا معمار اولیه تصمیم به عرضه محصول نرم افزاری گیرد. این کار، در روزهای اول پیاده‌سازی مدل بازار، بسیار مشکل خواهد بود. زیرا این مدل، با اکثر ساختارهای مدیریتی منافات دارد و باعث اعتراضاتی در

سازمان خواهد شد. اما با ارائه اولین محصول به بازار این فشارها کاوش خواهد یافت، اما هیچ‌گاه متوقف نخواهد شد.

۵-۹- تیم فناوری

بر اساس ساختار، تیم فناوری از معمار اولیه، معمار عرضه محصول و مهندسان تشکیل شده است. این تیم بخش دیگری هم دارد که بعداً درباره آن صحبت خواهد شد و آن تیم «بِر به بازار» است. وظیفه تیم فناوری این است که اصل فناوری را تولید کند و وظیفه آماده‌سازی آن برای بازار را تیم «بِر به بازار» بر عهده دارد. فعلاً نقشه‌های اساسی تیم فناوری و برخی از خروجیهای آن را بررسی خواهیم کرد.

۱-۵-۹- معمار اولیه

معمار اولیه باید ناظر اصلی باشد. این سمت تا حدی با مدیریت فناوری (CTO) در شرکتهای بزرگ فرق دارد. یک CTO در مقابل تصویر کلی فناوری یک شرکت مسئول و پاسخگوست. ولی معمار اولیه فقط تصویر فناوری محصول خاص را در نظر دارد. مثلاً با فرض پروژه پیاده‌سازی سامانه‌عامل لینوکس معمار اولیه باید بداند که هدف اصلی از این پروژه چیست: کارایی، قابلیت گسترش، توانایی بیشتر یا شبکه. اگر همه اینها اهداف پروژه هستند، اولویت‌بندی آنها چگونه است؟ چگونه می‌توان یک هدف را فدای دیگری کرد؟

برخی عقیده دارند که مختصصان فناوری قادر نیستند نیازهای مشتریان و الزامات بازار را درک و در سازمان اعمال کنند. اما جامعه متن‌باز اثبات کرده است که این ادعا اشتباه است. توجه شما را به این حقیقت جلب می‌کنم که یکی از مهمترین انگیزه‌ها برای هر مهندس نرمافزاری این است که بتواند محصول خود را به عموم عرضه کند. افرادی هم وجود دارند که مسخ زیباییهای فناوری می‌شوند. اما این افراد در اقلیت هستند. آنها که فریب زیباییهای فناوری را نمی‌خورند، معماران آینده شما خواهند شد.

باید معمار اولیه، خود را به کمک چارچوبهایی کلیدی هدایت کند تا تعادل خوبی بین نوآوری و بی‌رقیب بودن محصول بقرار شود. چارچوبهایی که شما باید مناسب با سازمان خود در نظر بگیرید عبارتند از:

- آیا محصول تولید شده قابل رقابت با هم نوعان خود در بازار می‌باشد؟
 - آیا این محصول اهداف ارضاء کننده مشتریان (امکانات، کیفیت، کارایی) را برآورده می‌کند؟
 - آیا فناوری در حال توسعه، خلاقانه است؟ آیا نقاط قوتی در این محصول وجود دارد که شما را از سایر تولیدکنندگان متمایز کند؟
- این درست نیست که اجازه دهیم معمار اولیه به دلخواه خود جهت‌گیری کند. او نیاز به کمک دارد و اینجاست که تیمهای بازاریابی وارد می‌شوند. در مجموع باید حداکثر اطلاعات را در اختیار او قرار دهید تا بتواند بهترین تصمیم را بگیرد. یک معمار باسابقه، خودش برای بدست آوردن این اطلاعات تلاش خواهد کرد.

معمار اولیه باید مرز زیرسامانه‌ها را نیز تعیین کند. هر پروژه نرمافزاری بزرگ، از زیرسامانه‌هایی تشکیل شده است. این زیرسامانه‌ها به گونه‌ای باید تعیین شوند که حداقل وابستگی بین آنها وجود داشته باشد. از آنجایی که وابستگی به این زیرسامانه‌ها اجتناب‌ناپذیر است، معمار باید واسطه بین آنها را معین کند. با روشن شدن مرز بین سامانه‌ها، معمار اولیه می‌تواند معماری هر یک از آنها را به شخص مورد اعتماد خود واگذار کند.

معمار اولیه همچنین در مقابل کدهای عرضه شده توسط معماران زیرسامانه‌ها نیز مسئول است. از آنجایی که چارچوبهای مشخص شده، معمار اولیه را در مقابل محصول نهایی پاسخگو می‌داند، او مجبور است از معماران قوی برای معماری زیرسامانه‌ها استفاده کند. مدیریت استعدادها از همین‌جا آغاز می‌شود بدین ترتیب که سلسله مراتبی از اعتماد بین افراد ایجاد می‌شود. اگر زیر سامانه‌ای انتظارات را برآورده نکند، معمار جدیدی برای آن معرفی می‌شود. معمار اولیه باید چنان مقتدرانه عمل کند که بتواند به راحتی معمار جدید را جایگزین کند. اگر این انعطاف از دست برود، به همان مدل ستی کاتدرال باز خواهد گشت. درجه حساسیت نسبت به کارهای ضعیف باید بسیار بالا باشد، یعنی از قبول چنین کارهایی به طور جدی جلوگیری شود. از وظایف معمار اولیه این است که با در نظر گرفتن پیامدهای عرضه محصول بی‌کیفیت، سعی کند محصولی بی‌رقیب، در زمان معین، تهیه کند.

معمار اولیه باید تشخیص دهد کی زمان عرضه محصول فرا رسیده است. تشخیص وی باید بر اساس عواملی از قبیل کیفیت، امکانات و قابلیتهای محصول استوار باشد. در نهایت اوست که در مقابل پیامدهای آن مسئول شناخته می‌شود. اگر عرضه محصول قبل از موعد واقعی آن باشد، آنگاه قابل رقابت نخواهد بود و اگر دیرتر از آن صورت گیرد، رقیبان بازار را از آن خود خواهند کرد. این عوامل متضاد، وی را به تشخیص بهترین زمان عرضه محصول راهنمایی خواهد کرد. شما به عنوان یک مدیر، باید پاداشها و جریمه‌های مناسبی برای او در نظر بگیرید تا انگیزه کافی برای تصمیم‌گیری بهتر داشته باشید. این پاداشها و جریمه‌ها نباید دروغین یا غیره واقعی باشند. در غیر این صورت نتیجه عکس خواهد داد. در عین حال نباید به گونه‌ای عمل کنید که گویا او تنها نابغه در این سامانه است. معمارها باید نابغه‌های تیم فناوری را از دیگران تمیز دهند و با جرأت از آنها بهره گیرند.

نکته آخر اینکه معمار اولیه باید رابطه دو طرفه‌ای را با معمار عرضه محصول، برقرار و حفظ کند. وقتی که محصول آماده شد، معمار عرضه وظیفه دارد که اصلاحات و بهبودهای جزئی را روی آن انجام دهد. از آنجایی که معمار عرضه در موقعیتی قرار دارد که تصمیم‌گیریهای او طراحی و معماری آینده کل سامانه را تحت تأثیر قرار می‌دهد، باید اعتماد دو طرفه‌ای بین او و معمار اولیه وجود داشته باشد. ارتباط این دو معمار شبیه رابطه‌ایست که لینوس و مارچلو هنگام طراحی هسته لینوکس با یکدیگر داشته‌اند.

۲-۵-۹ - معمار عرضه

معمار عرضه را باید معمار اولیه‌ای دانست که در حال انتظار است. همچنین می‌توان او را به عنوان «ستاند جبران حادث غیر متربقه» تلقی کرد. با توجه به اینکه ممکن است شرایطی پیش آید که مجبور به جایگزینی معمار اولیه با شخصی دارای مهارت‌های مورد نیاز شوید، معمار عرضه می‌تواند جای خالی او را موقتاً پر کند و از بحران در روند پروژه جلوگیری کند. بسیاری این سؤال را پرسیده‌اند که «اگر لینوس توروالدز را اتوبوس زیر بگیرد چه بلای سر لینوکس خواهد آمد؟» جامعه متن‌باز یک سامانه بازگشت به شرایط امن را تعییه کرده است. در صورت چنین اتفاقی، مارچلو می‌تواند بالاصله جایگزین لینوس شود و حرکت لینوکس را ادامه دهد. گرچه ممکن است در آینده جامعه متن‌باز شخص دیگری را برای راهبری لینوکس مناسب تشخیص دهد.

صرف‌نظر از شرایط فوق، بعد از اینکه معمار اولیه محصول را به سازمان واگذار می‌کند، فعالیت معمار عرضه شروع می‌شود. هر چند کیفیت محصول بالا باشد، باز هم اشکالات یا نقاط ضعفی در آن وجود خواهد داشت. در بعضی از سامانه‌ها ممکن است خلل‌های امنیتی وجود داشته باشد که باید به سرعت برطرف شود. معمار عرضه مسئول پاسخگوئی به مشکلاتی است که بعد از عرضه نرم‌افزار آشکار می‌شود. در عین حال نباید کلیت معماری سامانه را تغییر دهد. اگر پیشنهادهای بررسد که اعمال آنها نیازمند تغییر اساسی در معماری سامانه باشد، در این صورت این پیشنهادها باید به معمار اولیه منتقل شود تا او در زمینه اعمال آنها در نسخه‌های بعدی تصمیم‌گیری کند.

همان گونه که در نمودار سازمانی می‌بینید، معمار عرضه نیز با معماران زیرسامانه‌ها در تعامل است. نکته حائز اهمیت این است که باید نقاط کنترل مشترکی برای هر زیرسامانه وجود داشته باشد. در ادامه نقش معماران زیرسامانه‌ها را بررسی خواهیم کرد. سپس در این زمینه بحث خواهیم کرد که چگونه معمار عرضه می‌تواند تیمها را تشویق به حل مشکلات پیش آمده در نرم‌افزار عرضه شده، نماید.

به دلایل متعددی می‌توان گفت که وظایف معمار عرضه بسیار سنگین‌تر از معمار اولیه است. معمار عرضه باید به اندازه معمار اولیه، از معماری سامانه آگاهی داشته باشد و در عین حال باید به تغییرات بازار نیز واقف باشد. به طور خلاصه معمار عرضه کپی معمار اولیه است، با این تفاوت که حق ایجاد تغییر اساسی در معماری سامانه را ندارد.

چارچوب کاری معمار عرضه بر حفظ کیفیت و پایداری محصول استوار است. معمار عرضه باید بتواند در مدت کوتاهی برنامه‌های اصلاحی نرم‌افزار عرضه شده را آماده کند. در این صورت شما نگرانی از بابت کیفیت نخواهید داشت. اگر معمار عرضه اطلاعات درستی از کلیت سامانه نداشته باشد، در انجام وظیفه پاسخگویی به مشتریان به مشکل بر می‌خورد. از طرف دیگر اگر او نتواند رابطه مورد نیاز را با تیمها برقرار کند، قادر نخواهد بود تغییرات مورد نیاز را به سامانه اعمال کند. با نظرسنجی درباره کیفیت محصول می‌توان از میزان رضایت مشتریان و نحوه عملکرد معمار عرضه مطلع شد.

۳-۵-۹ - معماران زیرسامانه‌ها

معماران زیرسامانه‌ها در حقیقت سلسله مراتبی از معماران یا طراحان سامانه هستند. عمق این سلسله مراتب بستگی به پیچیدگی سامانه دارد. در این مبحث عمق سلسله مراتب را یک لایه در نظر می‌گیریم. اگر لایه‌ها افزایش یابد، معمار سامانه بالادست نقش معمار اولیه را برای سامانه پایین دست ایفاء می‌کند.

اگر مروری به فناوری سامانه‌عامل لینوکس بیاندازید، به تعدادی زیرسامانه برمی‌خورید: حافظه مجازی، سامانه‌های پرونده، مدیریت پرونده‌ها، شبکه و غیره. همان گونه که قبلاً هم اشاره شد مرز میان این زیر سامانه‌ها توسط معمار اولیه تعیین می‌شود و ممکن است در روند تکامل سامانه، این مرزها تغییر کند. معمار سامانه حتی الامکان نباید پارامترهای تعیین شده برای زیرسامانه را تغییر دهد. اما می‌تواند بر تصمیم‌گیریهای معمار اولیه تأثیر گذارد.

معمار زیرسامانه در مقابل طراحی زیر سامانه خود مسئول است. او کارهای انجام شده توسط مجموعه مهندسین تحت امر خود را دریافت و ارزیابی می‌کند. او باید با گروه منابع انسانی نیز در تماس باشد و از موجود بودن افراد با استعداد، در صورت نیاز احتمالی، اطمینان حاصل کند. این انگیزه در مهندسان وجود دارد که کارهای خود را به چندین زیرسامانه عرضه کنند تا بتوانند در سلسله مراتب ایجاد شده، ارتقاء یابند. اما این انگیزه را باید معماران سامانه‌ها در آنها ایجاد کنند و این ضروریست.

وقتی که صحبت از ایجاد یک فناوری جدید می‌شود، معماران زیر سامانه‌ها باید سامانه‌ای با کیفیت و عملیاتی به معمار اولیه (یا معمار زیر سامانه بالادست خود) ارائه کنند. معمار اولیه باید چنان معماران زیر سامانه‌های خود را قبول داشته باشد که نیازی به بازنگری کارهای آنها نداشته باشد.

یک معمار زیرسامانه وظایفی نیز نسبت به معمار عرضه دارد. هر اشکالی که در زیرسامانه وجود داشته باشد از طریق معمار عرضه دریافت می‌شود. معماران زیرسامانه‌ها باید به تعامل با معمار عرضه تشویق شوند تا هر چه بهتر و سریعتر اصلاحات مورد نیاز را انجام دهند. هر چه واکنش به مشکلات بهتر صورت گیرد، کیفیت کل سامانه بالاتر می‌رود.

برخی از چارچوبهایی که شما می‌توانید برای معماران زیرسامانه‌ها قرار دهید عبارتند از:

- توانایی جذب استعدادها به سمت زیرسامانه خود- به علاوه خود شما نیز باید از موجود بودن استعدادهای مورد نیاز اطمینان حاصل کنید.
- تحويل خروجی با کیفیت و به موقع- این مسئله را معمار زیرسامانه بالاتر باید تشخیص دهد. اگر خروجی زیرسامانه‌ای با اهداف مورد نظر منطبق نباشد، معمار زیر سامانه باید عوض شود.
- نوآوری در زیرسامانه- فناوریهای جدید و ابداعات باید در کل سامانه صورت گیرد.
- تعیین جانشین- همان گونه که گفتیم معماران عرضه و اولیه، یک سامانه «جبران حوادث غیرمنتقبه» تشکیل می‌دهند، اما معماران زیرسامانه‌ها لزوماً چنین سامانه‌ای

نارند. هرچند بهتر است چنین سامانه‌ای را خودشان تعییه کنند. در صورتی که معمار زیرسامانه‌ای قصد داشته باشد در سلسله مراتب ارتقاء پیدا کند، باید جانشینی برای خود تعیین کرده باشد. بدین ترتیب سامانه جراث حادثه را نیز تشکیل داده است.

همان گونه که مشاهده می‌کنید، طی این فرآیندها شما یک فرهنگ جدید در سازمان خود ایجاد می‌کنید. به جای اینکه سلسله مراتب سازمان به گونه‌ای باشد که تصمیم‌گیریها از بالا به پایین صورت گیرند، فرهنگی تکامل می‌یابد که بر اساس رشد مهندسان و تقاضای آنها برای انجام دادن کار بهتر استوار است.

۴-۵-۹ - مهندسان

مهندسان سازمان شما، دانش شما هستند، آنها دارایی گرانبهای شما هستند و اجزای قابل تعویض نیستند. وقتی که مهندسی وارد سازمان می‌شود، نیاز به زمان دارد تا بتواند خود را با فرهنگ سازمان وفق دهد و سامانه در حال طراحی را بفهمد. افراد مختلف ممکن است اهداف متفاوتی داشته باشند. برخی می‌خواهد تا بالاترین سطح سلسله مراتب رشد کنند، برخی دیگر می‌خواهند با انجام دادن کار بیشتر دانش خود را گسترش دهند. عده‌ای نیز تمایل دارند در قله یک فناوری خاص قرار گیرند. شما باید ارزش هر یک از این سه گونه از مهندسان را تعیین کنید. همه آنها به ارزش سامانه می‌افزاید.

برای اینکه یک مهندس بتواند در سلسله مراتب ارتقاء یابد، باید بتواند علاوه بر استعداد فنی، تا حدی مهارت معماری خود را نیز نشان دهد. حتی مهارت‌های روابط عمومی فرد نیز در این زمینه نقش به سزاپی دارد. ارزش مهارت‌های این افراد هر چه در سلسله مراتب بالاتر می‌روند، بیشتر می‌شود.

استعداد مهندسی افراد، باید از طریق کیفیت، کمیت، تنوع و نوآوری کارهای آنها سنجیده شود. اگر زیرسامانه‌ای به نقطه خاصی از رشدیافتگی برسد، مهندسین آن زیرسامانه یا باید خلاقلیت خود را در آن زیرسامانه نشان دهند، یا به زیرسامانه‌های دیگر منتقل شوند. اگر کمیت کارها کاهش یابد، باید آن استعداد را با دلگرمی دادن اجرا کرد.

معیار مهمی که می‌توان بر آن تکیه کرد، تعداد کارهای مردود مهندسان است. از آنجایی که تمامی سطوح سلسله مراتب در مقابل کیفیت کلی سامانه مسؤول هستند، کارهای ضعیف و بی‌کیفیت با قاطعیت رد می‌شوند. برخی از کارهای مردود، طبیعی و قبل از این ارائه کارهای مردود به طور مکرر، نشانگر وجود مشکلی در میان مهندسان است که باید برطرف شود.

گروه منابع انسانی به طور پیوسته باید استعداد مهندسی سازمان را بهبود دهد. این گروه را با جزئیات بیشتری بررسی خواهیم کرد. دقت کنید که باکس مهندسان ضمایمی هم دارد. این ضمایمه‌ها در یک سازمان بزرگ نقش کلیدی دارند. یکی از مزایای مهم روش توسعه بازار این است که همه می‌توانند وارد بازی شوند. در عین اینکه کارمندان شما باید از آموزش کافی در زمینه ساختار سازمان برخوردار باشند، فرهنگی که ایجاد می‌کنید نیز باید به گونه‌ای باشد که بتواند از خارج سازمان نیز نیروی کار جذب کند. اگر روزی دیدید که بخش‌های دیگر سازمان نیز

مایل به انجام کاری در زمینه بهبود سامانه هستند، بدانید که قدم بزرگی در رشد فرهنگ سازمان خود برداشته‌اید. چنین شرایطی اولاً علامت این است که سامانه کنونی پاسخگوی نیازهای کاربران است، ثانیاً علامت آن است که شما نیروهای مستعد و با انگیزه جدیدی در اختیار دارید که می‌توانند سامانه را بهبود دهند. کارهای این گروه نیز باید توسط یکی از معماران ارزیابی و سپس وارد سامانه شود. طبیعتاً کارهای ضعیف باید رد شوند. این باعث می‌شود سایر کارمندان شرکت نیز بدانند که اگرچه تشویق به کمک به بهبود سامانه می‌شود، اما فقط کارهای با کیفیتی که در چارچوبهای معماري سامانه می‌گنجند، مورد قبول واقع می‌شوند.

جای دیگری که ممکن است نیروی مستعد وجود داشته باشد، مشتریان و شرکاء شما هستند. این بخش در طبقه‌بندی جوامع دروازه‌ای می‌گنجد که در همین فصل بررسی خواهد شد.

۹-۶- منابع انسانی

جامعه متن‌باز اصولاً خود را زیاد درگیر هزینه‌های نیروی انسانی نمی‌کند. آنها لزومی نمی‌بینند که در زمینه پادشاه، کشیدن چک و تهیه تجهیزات نگرانی داشته باشند. گروه منابع انسانی، از موارد اصلی اختلاف بین روش توسعه بازار در جامعه و شرکتهاست.

نقشهای وظایف گروه منابع انسانی که در اینجا شرح داده خواهد شد، ممکن است فراتر از آنچه باشد که شما از آنها انتظار دارید. نقش این تیم بسیار حساس است. هدف گروه فناوری این است که تمام انرژی خود را بر فناوری و آماده کردن محصول مرکز کند. بدین ترتیب مدیریت درست کارمندان در این راستا بسیار مهم است. این وظیفه را یا باید خود تیم بر عهده گیرد یا به تیم منابع انسانی واگذار کند.

۹-۶-۱- رشد استعدادها

نیروی انسانی برای رشد استعدادهای سازمان باید منابع مورد نیاز را در اختیار داشته باشد. ممکن است لازم باشد کارمندان از دوره‌های آموزشی خارج سازمان استفاده کنند یا در یک ناحیه خاصی کارآموزی کند. در شرکتهای سنتی رشد افراد بر عهده مدیر مستقیم آنها می‌باشد. در این روش مدیریت مستقیم دیگر وجود ندارد، اما همچنان منابعی برای رشد افراد مورد نیاز است.

ایجاد ارتباطات داخل سازمانی نیز، برای رشد کارمندان لازم است. همچنان مدیریت شرکت باید فرآیندی را برای انتقال سیاست‌گذاریهای شرکت به کارمندان در نظر گیرد. بدین ترتیب علاوه بر اینکه کار روزانه گروه مهندسی شرکت منطبق با اهداف فناوری در حال توسعه می‌گردد، ارتباطات داخل شرکت باعث می‌شود کارمندان از اهداف استراتژیک شرکت نیز آگاهی یابند.

۹-۶-۲- مدیریت افراد (مهارت‌ها)

این گروه وظیفه دارد که نیازهای مهارتی معماران را برآورده کند و از موجود بودن نیروی ماهر اطمینان حاصل کند. اگر مهارت کمیابی مورد نیاز باشد، باید در جدب این مهارت سرمایه‌گذاری خوبی شود. از وظایف دیگر این تیم این است که حتی‌الامکان طیف گسترده‌ای از انواع مهارت‌ها را آماده به کار داشته باشد. منظور از طیف گسترده مهارت‌ها این است مخلوطی از افراد ماهر

عالقمند به فناوری خاص است که در سلسله مراتب مدیریتی رشد می‌کنند و به درجات مختلفی از مدیریت می‌رسند.

این تیم باید توانایی مالی شرکت را نیز در استخدام کارمندان مدنظر داشته باشد. این کار را می‌توان از طریق استخدام کارمندان قراردادی و دائم محقق کرد. اما اگر شرکت قادر نباشد، نیروی جدیدی استخدام کند. وظیفه این تیم است که با گروههای مهندسی مذاکره کند و نیروهای ماهرتر را به زیرسامانه‌ای که نیاز بیشتری دارد، منتقل کند. در این شرایط ممکن است مهندسان به این نتیجه برسند که فعالیت خود را بر روی یک زیرسامانه متوقف کند و بر زیرسامانه مهمتری تمرکز کنند. اما عده‌ای هم ممکن است تصمیم به بیکار ماندن تا باز شدن مجدد آن زیرسامانه بگیرند که عواقب منفی آن متوجه خودشان خواهد شد.

قسمتهای دیگر مدیریت مهارت باید از موجود بودن ماهرترین افراد و رشد آنها اطمینان حاصل کند. این تیم باید فرآیندهای را برای محک زدن کار مهندسان تعییه کند. معیارهای سنجش کار مهندسان عبارتند از: سرعت، تنوع و میزان کارهای مردود. آنها باید با کم کاریهای کارمندان، به طور جدی برخورد کنند. گرچه این روش ممکن است خشن به نظر برسد، اما باعث ایجاد سامانه خود اصلاحی می‌شود و افرادی که نمی‌توانند خوب کار کنند، خودشان باید دنبال راه حل دیگری داخل یا خارج شرکت باشند. اینکه چه مدت لازم است تا فردی بتواند کار مفید و پیوسته ارائه دهد (با شرایط سازگار شود) بستگی به فرهنگ مرسوم در شرکت دارد.

۳-۶-۹ - آموزش

این تیم وظیفه آموزش کلی کارمندان و آموزش فرآیندی مهندسان را بر عهده دارد. آموزش کلی، اغلب ترکیبی است از آموزش فرهنگ کار و شرح اهداف و سیاستهای شرکت. آموزش فرآیندی مهندسان نیروهای ماهر را آماده کار در زیرسامانه‌ها می‌کند. در این آموزشها می‌توان فناوری در حال توسعه را تا سطوح پیشرفته آموزش داد، اما تمرکز اصلی باید بر ابزار مورد استفاده مهندسان و نحوه انجام کار باشد. اکثر مهندسان مایلند هر چه زودتر آموخته‌هایشان را در عمل اجرا کنند. آنها می‌توانند پس از یادگیری فرآیند کار، با تمرین و تعامل با سایر مهندسان، کار خود را شروع کنند. نقش تیم آموزش را دست کم نگیرید. اگر مهندسان از نحوه مؤثر انجام کار مطلع نباشند، همه ضرر خواهند کرد.

۷-۹ - آزمایش و یکپارچگی

این هم از دیگر وظایف مربوط به شرکت‌ها است که در جامعه متن‌باز به چشم نمی‌خورد. فرآیند آزمایش کردن و یکپارچه کردن نرم‌افزارها در جامعه متن‌باز به سادگی ارسال پرونده‌های منبع، وصله‌ها و اصلاحات از طریق پست الکترونیکی به نگهدارنده آن نرم‌افزار می‌باشد. مجموعه ابزاری که در فصل ۳ به آنها اشاره شد (CVS، Bugzilla و ...) نقاط اثکای مهمی برای این جامعه هستند. آزمایش کردن نرم‌افزار به این ترتیب انجام می‌شود که جمعیت عظیمی از کاربران در جاهای مختلف دنیا، حين کار با آن نرم‌افزار، اشکالات را پیدا می‌کنند. در ناسا به ازاء هر برنامه‌نویس از ۳۰ مهندس، برای آزمایش برنامه‌ها استفاده می‌شود. در حالی که در جامعه متن‌باز،

نرم افزارها رسماً آزمایش نمی شوند. سؤال اصلی این است که با این وجود چگونه جامعه متن باز توانسته است به نتایج کیفی قابل توجهی دست یابد. ضربالمثل ریموند را به خاطر آورید که «اگر به اندازه کافی چشم وجود داشته باشد، همه اشکالات برنامه ها (باگها) کم عمق می شوند» این است راز کیفیت نرم افزارهای متن باز.

در اغلب شرکتها دسترسی به کد منبع به شدت کترنل می شود و فقط مراجع خاصی (مثل دولت ها) قادر به نگاه کردن به آن هستند. همچنین تغییر دادن در کد منبع نیز برای عده اندکی امکان پذیر است. شما باید این فرهنگ را نهادینه کنید که هر چه افراد بیشتری کد منبع را ببینند، بهتر است. نظرات در زمینه طراحی، نحوه کدنویسی و الگوریتم های برنامه را باید به گرمی بپذیرید، اما به خاطر داشته باشید که در نهایت شما می خواهید محصول را به صورت تجاری عرضه کنید. بنابراین کاربران شما برای آزمون کردن و کترنل کیفیت محصول روی شما حساب می کنند و از آنجایی که نسبت برنامه نویسان به کاربران در مورد محصول شما بسیار کمتر از موردهای طراحی شده در جامعه متن باز است، لازم است محصول خود را به دقت آزمایش کنید.

در فصل دوازدهم در زمینه فرآیند در دسترس قرار دادن کد منبع، تحت یک مجوز متن باز بحث خواهیم کرد. معمولاً پیش بینی زمان این کار امکان پذیر نیست. بهتر است برای یکپارچه سازی کد منبع و انتقال آن به جامعه متن باز از همان ابزاری استفاده کنید که آنها استفاده می کنند. در برخی موارد قوانین و محدودیتهای شرکت، ممکن است شما را مجبور به استفاده از سامانه های یکپارچه سازی خاصی کنند. در چنین شرایطی باید به شرکتهایی مراجعه کنید که ابزارهایی برای ترکیب متن باز با روشهای سنتی ارائه کرده اند.

با این همه شما باید ساختار آزمون محصول را حفظ کنید. اگر فرهنگ جدید درست پیاده شده باشد، باید شاهد کاهش قابل توجه مشکلات یافت شده در هنگام آزمون محصول باشید. مدیران باتجربه باید بدانند که هدف از ایجاد تیم آزمون فقط این نیست که اشکالات را آشکار کند، بلکه کیفیت کار کارمندان را نیز معنکس می کند.

۱-۷-۹ - گروه مالی

همکاران امور مالی شما نیز همان وظیفه سنتی خود را بر عهده خواهند داشت، اما با چند تفاوت جزئی. وقتی که افراد دارای مهارت های مختلف را استخدام می کنید پایه سامانه دقیق برای حقوق و دستمزد آنها تعییه کنید. گروه مالی باید نیازهای تیمهای فناوری و «بیرونی بازار» را نیز در تعادل نگه دارد و در عین حال باید با گروه متابع انسانی نیز در ارتباط باشد تا هزینه نیروی انسانی از مقادیر پیش بینی شده فراتر نرود.

۸-۹ - بازاریابی

از دیگر وظایفی که در جامعه متن باز به چشم نمی خورد بازاریابی است. به یادآورید که گفتیم در جامعه متن باز نویسنده یک برنامه، استفاده کننده از آن نیز هست و این بسیار جالب است. اما وقتی که شما محصولی را برای داخل شرکت یا عرضه تجاری تهییه می کنید باید دید جامعه تری از نیاز کاربران داشته باشید. در روشهای بازاریابی رایج، ابتدا مشتریان دسته بندی و اولویت بندی

می‌شوند و سپس نیازهای آنها به ترتیب اولویت به عنوان اهداف شرکت در نظر گرفته می‌شود. بنابراین شما می‌توانید به روشی بازارهای هدف کنونی و آینده خود را برای معمار اولیه مشخص کنید. این اطلاعات راهنمای بسیار خوبی در راستای پیشرفت پروژه هستند و این مزیت کار شرکتی نسبت به کار انجام شده جامعه متن‌باز است و سود قبل توجهی عاید شرکت خواهد کرد. وظیفه دیگر این گروه این است که چارچوبهای کاری تیمها را تعیین و به طور پیوسته با توجه به شرایط بازار به روز کند و بدین ترتیب جهت‌گیری آنها را منطبق بر نیازهای کاربران حفظ کند. سنجش توانایی رقابتی محصول معمار خوبی از کیفیت کار معمار اولیه است. این گروه تأثیر جانبی دیگری نیز دارد. اکثر تیمهای بازاریابی مایلند مشخصات محصول را تا جزئیات ریز آن تعیین کنند. بنابراین تیمهای مهندسی احساس می‌کنند که جایی برای نوآوری و خلاقیت آنها در نظر گرفته‌اند. اما این تیم فقط بازارهای هدف و اهداف رقابتی مفید را معین می‌کند و مهندسان انگیزه زیادی برای تهیه فناوری مطلوب با امکانات مناسب خواهند داشت.

۹- بِر به بازار

من تا اینجا تلاش کردم عرضه فناوری را به گروه فناوری ربط ندهم. علت این است که وظیفه عرضه یک محصول آماده شده بر عهده «تیم بِر به بازار» است. برای درک این مسئله کافیست نگاهی به نحوه عرضه سامانه‌عامل لینوکس بیاندازیم. لینوس توروالدز به همراه تیم ماهری از برنامه‌نویسان، هسته لینوکس را تهییه می‌کنند. هزاران برنامه‌نویس دیگر نرم‌افزارهای سامانه GNU را تهییه می‌کنند و بهبود می‌دهند. سپس تعدادی از توزیع‌کنندگان لینوکس همه این اجزاء را با یکدیگر ترکیب می‌کنند و آنها را به صورت یکپارچه تحت یک محصول برای استفاده کاربران عرضه می‌کنند. آنچه که باید به آن اشاره شود این است که هر توزیع‌کننده‌ای، این اجزاء را به صورت خاصی جمع می‌کند که با نحوه جمجم‌آوری توزیع‌کننده دیگر متفاوت است. آنها ابزارها و قابلیتهایی را نیز به آن اضافه می‌کنند با توجه به اینکه هدف آنها چه بازاری باشد. همان گونه که قبلاً اشاره شد برخی از توزیع‌کنندگان ممکن است بازار کارسازها را هدف گرفته باشند و برخی دیگر بازار رایانه‌های رومیزی را. هدف برخی دیگر از توزیع‌کنندگان برنامه‌های ارتباطی بلاذرنگ می‌باشد.

این گروه نقش توزیع‌کننده را برای محصولات شرکت بازی می‌کند و می‌تواند کاری کند که شما بتوانید یک فناوری را به بخش‌های مختلف بازار عرضه کنید. اگر پروژه‌های شما کوچک باشند، ممکن است یک تیم بِر به بازار نیاز شما را برآورده کند. اما در زمینه پروژه‌های بزرگ مسلماً یک تیم کافی نخواهد بود. مدیریت شرکت باید مرز بین پایان فعالیت گروه فناوری و آغاز فعالیت گروه بازار را دقیقاً مشخص کند. مفهوم این جمله این است که شما در گروه بِر به بازار نیز تا حدی نیازمند مهارت مهندسی هستید. این مهندسان نیازمند مهارت‌هایی غیر از مهارت‌های مورد نیاز در تیم مهندسی از قبیل درک مسائل بازار هستند.

وظیفه دیگر این تیم است که زنجیره‌های ارزش محصول را کامل کند از قبیل تعامل با گروههای فروش، تولید، پشتیبانی و همه گروههایی که در عرضه یک محصول قابل استفاده، چه برای داخل سازمان و چه برای عرضه به مشتریان، نقش دارند.

۱۰-۹ - سایر اجزای ساختاری

این مبحث بیشتر بر نحوه انجام امور مهندسی و ارتباطات آن با سایر تیمها متمرکز شده است. تأثیر سامانه جدید بر سایر ارتباطات ساختاری نیز باید بررسی شود. این نکته حائز اهمیت است که بتوان تغییرات صورت گرفته در سازمان ناشی از روش مدیریت جدید را به گروه پشتیبانی، تولید، فروش و سایر بخش‌های کلیدی سازمان منتقل کرد. به عنوان مثال گروه پشتیبانی از اینکه می‌تواند در توسعه و پیشرفت پژوهه نقش داشته باشد، استقبال خواهد کرد و ممکن است ایده‌های خوبی نیز در این زمینه ارائه دهد. توسعه از بخش‌های دیگری نیز تشکیل شده است که متناسب با شرایط سازمان، باید به آنها پردازد. به عنوان مثال ممکن است شما برای اینکه از میزان کارایی محصول خود آگاه شوید یک تیم عوامل انسانی نیز به ساختار سازمانی اضافه کنید که با گروه مهندسی در تعامل باشد. به طور خلاصه نکته کلیدی در این روش، برقراری و حفظ ارتباط مؤثر بین اجزاء سازمان است.

۱۱-۹ - جوامع دروازه‌ای

همچنان که فرهنگ سازمان شما رشد می‌کند و بازیگران بیشتری به جامعه توسعه‌دهندگان شما اضافه می‌شود، به نقطه‌ای خواهید رسید که مشتریان و شرکاء‌ی کلیدی شما نیز علاقمند خواهند شد که به مشارکت فرآیند توسعه دخیل می‌گردند. از آنجایی که نمی‌توان همه را برای مشارکت در این فرآیند راه داد، باید این همکاری را به صورتی قانونمند درآورید.

فراموش نکنید که هدف این فصل این است که فرآیند توسعه بازار را در سازمان پیاده کنید، نه اینکه کد منبع خود را در اختیار عموم قرار دهید. جوامع دروازه‌ای به فرآیند دخیل کردن مشتریان و شرکاء در امر توسعه در عین حفظ کردن کد منبع به صورت اختصاصی گفته می‌شود.

چه دلیلی دارد که مشتریان و شرکاء را در فرآیند توسعه دخیل کنیم؟ یکی از مزایای اصلی متن‌باز واکنش سریع به مشکلات است. در دنیای نرم‌افزارهای اختصاصی، مشتریان بزرگ وقتی که شرکت‌های تولیدکننده نرم‌افزار نمی‌توانند خواسته‌ها و نیازهای آنها را برآورده کنند، نالمید می‌شوند. برخی از این خواسته‌ها در حد رفع یک اشکال جزئی است و برخی دیگر ممکن است در حد افزودن امکانات پیچیده به نرم‌افزار باشد. شرکاء به شرکت‌های اطلاق می‌شود که محصولشان متکی به محصول شماست. دخیل شدن شرکاء نیز باعث بهبود کیفیت محصول شما خواهد شد. به همین دلیل بزرگترین بهره‌ای که از جوامع دروازه‌ای می‌برید، نوآوریهاییست که نیازهای مشتریان شما را برآورده می‌کند و پاسخگوئی به نیازهای مشتریان را سریعاً بهبود می‌بخشد. ایجاد جوامع دروازه‌ای نیازمند تشكیل زیرساخت شبکه‌ای است که به افراد خارج از سازمان اجازه دسترسی به کدهای منبع را بدهد و در عین حال امنیت کار شما را نیز تأمین کند.

همچنین باید کنترل‌هایی نیز برای شناسائی شرکای اصلی که حق دخالت در پروژه را دارند، وجود داشته باشد.

شما باید قراردادهایی نیز در زمینه حفظ محرمانگی استناد با شرکاء منعقد کنید. اگر ساختار کد منع را بر اساس قواعد متن باز بنا کنید، متوجه خواهید شد که شرکاء خواهند توانست به طور مؤثری با جامعه برنامه‌نویسان پروژه شما همکاری کنند. همچنین باید توافق نامه‌ای در زمینه مالکیت معنوی کارهایی که از طرف مشتریان و شرکاء ارائه می‌شود، تنظیم کنید. اگر کدهای منبع را تحت یکی از مجوزهای متن باز عرضه کنید، در این صورت کمترین محدودیتها را برای آن قائل شده‌اید.

۱۲-۹ - خطرات و مشکلات

هر ساختار جدیدی با خطرات و نقصهایی مواجه است و ساختار بازار نیز از این قاعده مستثنی نیست. در ابتدای امر شما باید این ساختار عملیاتی را به یک ساختار سازمانی خاص سازمان خود تبدیل کنید. انتخاب بازیگران اصلی این ساختار تأثیر به سزاگی بر موفقیت این روش دارد. در فصل سیزده فرآیند انتخاب افراد و چارچوبها را با جزئیات بیشتری بررسی خواهیم کرد. انتقال از یک ساختار قدیمی به یک ساختار جدید نیز، خطراتی در پی دارد. اگر نقشه مهاجرت دقیق و با در نظر گرفتن جزئیات تهیه کرده باشید که ارتباطات نیز در آن در نظر گرفته شده باشد، در این صورت می‌توانید بر بسیاری از خطرات غلبه کنید. همچنین زمان پیاده‌سازی روش توسعه جدید در چرخه عرضه محصول نیز باید با دقت تعیین شده باشد. همان گونه که چندین بار اشاره کرده‌ام، شما باید فرهنگی متن باز در داخل سازمان پیاده کنید که کارمندان کمترین محدودیت را برای دسترسی به کدهای منبع داشته باشند.

احتمالاً بزرگترین خطر مربوط، به نحوه مدیریت کارمندان می‌شود. از آنجایی که این ساختار نقش مدیریت مستقیم بر گروه‌های مهندسی را کم رنگ‌تر می‌کند، باید از مشاوران و مریبان به طور مؤثرتری استفاده کنید و گروه منابع انسانی نیز باید ارتقاء سطح مهارت مهندسان را بالاترین اولویت خود قرار دهد.

۱۳-۹ - جمع‌بندی

در این فصل پیش زمینه‌های گسترهای در زمینه موضوعی بسیار پیچیده در اختیار شما قرار گرفت. ممکن است حتی لازم باشد این فصل را چندین بار مطالعه کنید. در این فصل ما نحوه عملکرد جامعه متن‌باز در یک دنیای تجاری را بررسی کردیم. امیدواریم شما علاوه بر درک نحوه رسیدن جامعه متن‌باز به اهداف خود، توانسته باشید برخی از روش‌های آنها را در سازمان خود به کار گیرید.

قبل از اینکه دانش شما را در زمینه توسعه متن‌باز به عمومی کردن کد منبع گسترش دهیم، میانبری می‌زنیم به تأثیرات متن‌باز بر مدل‌های تجاری رایج. این بحث دلایل متن‌باز کردن برخی از پروژه‌ها و دخیل کردن جامعه متن‌باز در محصولات و پروژه‌ها را روشن خواهد کرد.

فصل دهم

ارزش محصول به عنوان تابعی از زمان

فروشنده‌گان نرمافزار به این امید به تولید فناوری می‌کنند که درآمد فروش آن بیشتر از هزینه تولید آن گردد. هر چه نرمافزار منحصر به فرد و ارزش آن بیشتر باشد، قدرت قیمت‌گذاری فروشنده بیشتر می‌شود. عوامل اولیه تأثیرگذار بر قدرت قیمت‌گذاری محصول عبارتند از: ارزشی که خریدار برای محصول قائل می‌شود و فشار رقابتی موجود در بازار. متن باز با روشهای دیگری، قدرت قیمت‌گذاری را تحت الشاعع قرار می‌دهد و ارزش برخی از فناوریها را سریعتر پایین می‌آورد. در این فصل روشهای جدید رقابت در این بازار بررسی می‌شود. گرچه این فصل برای فروشنده‌گان نرمافزار به صورت تجاری نوشته شده است، اما برای مدیران فناوری اطلاعات نیز از این جهت مفید است که عوامل تأثیرگذار بر فروشنده‌گان نرمافزار را درک می‌کند و دید بهتری نسبت به روشهای رقابتی جدید در حال گسترش در بازار پیدا می‌کنند. درک نحوه تأثیرگذاری متن باز بر منابع سازمان نیز برای مدیران فناوری اطلاعات حائز اهمیت است. اهداف این فصل این است که دید بهتری نسبت به موارد زیر پیدا کنید:

- تأثیر اقتصادی منحنی ارزش-زمان
- تأثیرات متن باز بر ارزش نرمافزارهای تجاری
- تأثیر متن باز بر حفظ ارزش نرمافزار
- چگونه از تأثیرات متن باز بر منحنی ارزش-زمان بهره‌برداری شود

به عنوان یک تولیدکننده نرمافزار تجاری، شما باید نرمافزاری تولید کنید که ارزش آن بالاتر از ارزشمندترین نرمافزار موجود در بازار باشد، در غیر این صورت درآمد شما در حدی نخواهد بود که بازگشت سرمایه را تضمین کند. برای اینکه به درک بهتری از تأثیرات نرمافزارهای متن باز بر کسب و کار برسید نگاهی می‌اندازیم به تحولات بازار یکی از صنایعی که مدتی تحت تأثیر آن بوده است.

۱-۱۰- صنعت داروسازی

صنعت داروسازی سالهای است که درگیر مشکل کاهش ارزش محصولات پس از طی زمانی مشخص می‌باشد. در اواسط دهه ۱۹۸۰ من از درد معده رنج می‌بردم بعد از مدت حدود ۱۸ ماه این درد چنان شده بود که پس از هر بار غذا خوردن مرا آزار می‌داد. دکترهای زیادی مرا معاينه کردند، گرچه در آن زمان معالجه قطعی برای این مشکل وجود نداشت، اما می‌شد درد آن را با مصرف برخی داروها خفیفتر کرد. متأسفانه بدن من نسبت به این دارو واکنش نشان می‌داد و قادر به مصرف آن نبودم. در همان زمان داروی جدیدی به نام «رانیتیدین» به بازار آمد که امروزه به نام «زانتاک» نیز شناخته می‌شود. بعد از مصرف این داروی جدید به مدت یک هفته تمامی علائم بیماری من برطرف شد و به ۱۸ ماه درد کشیدن من پایان داده شد. من حاضر بودم این دارو را به هر قیمتی تهیه کنم. از نظر من این دارو یک معجزه بود. خوشبختانه من تحت پوشش بیمه خوبی بودم. هر روز باید ۱۲۰۰ میلی‌گرم از این دارو مصرف می‌کردم. با توجه به

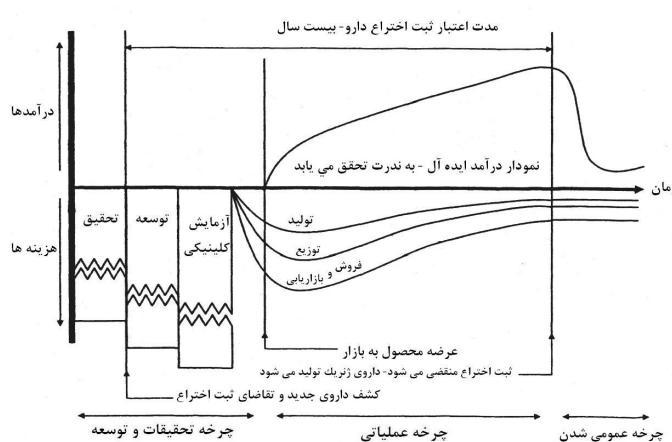
اینکه قیمت هر قرص ۱۵۰ میلی‌گرمی ۱/۸۰ دلار بود، مصرف روزانه من ۱۵ دلار در روز یا ۵ هزار دلار در سال می‌شد. از آنجایی که مصرف هر داروی دیگری وضعیت مرا بدتر می‌کرد، شرکت بیمه کننده چاره‌ای جز پرداخت این هزینه نداشت. در آن زمان هیچ رقابتی در زمینه رانیتیدین وجود نداشت. من به مدت هفت سال این دارو را مصرف کردم و باید اعتراف کنم که اگر سیگار نمی‌کشیدم، مدت درمان کوتاه‌تر می‌شد.

اخیراً قیمت رانیتیدین را پرسیدم و متوجه شدم که قیمت هر قرص ۱۵۰ میلی‌گرمی آن کمتر از ۱۲ صدم دلار است. درمان هفت ساله ۳۵ هزار دلاری من امروزه کمتر از ۲۵۰۰ دلار می‌شد. یعنی هزینه درمان من ۹۳ درصد کاهش یافته است. آیا می‌توانید تصور کنید که محصول شما به این سرعت افت قیمت پیدا کند. علت کاهش قیمت شدید رانیتیدین این بود که نوع ژنریک آن به بازار آمد. داروهای ژنریک پس از اینکه حق ثبت تولیدکننده اصلی آن منقضی می‌شود، به بازار می‌آید. در مورد داروهای مفیدتر نسخه ژنریک آنها در همان روز انقضاء حق ثبت عرضه می‌شود. اکنون باید بینیم صنعت داروسازی چگونه بر هزینه‌های خود فائق آمده است و مدل تجاری خود را چگونه تنظیم کرده است. سپس به صنعت نرم‌افزار باز می‌گردیم و تجربه صنعت داروسازی را در آنجا اعمال می‌کنیم.

۱-۱-۱- هزینه، ارزش، بازگشت سرمایه و زمان

صنعت داروسازی از این واقعیت بهره می‌برد که زمان دقیق انقضاء حق ثبت یک دارو مشخص است و همچنین پیامد آن که عرضه داروهای ژنریک به بازار می‌باشد، واضح است. این مسئله به صنعت داروسازی این امکان را می‌دهد که مدل هزینه به درآمد را دقیقاً بنا کند.

نگاهی به نمودار شکل ۱-۱۰ می‌اندازیم.



شکل ۱-۱۰ مدل بازگشت سرمایه تولیدات دارویی

در این نمودار چرخه اقتصادی داروهای جدید به زبان ساده نمایش داده شده است. من قصد ندارم شما را متخصص مدل‌های اقتصادی این صنعت کنم، بلکه می‌خواهم اثرات آن را با اثرات متن‌باز بر صنعت نرم‌افزار مشابه‌سازی کنم.

شرکتهای داروسازی هزینه زیادی را صرف تحقیقات برای تولید داروی جدید می‌کنند. هدف آنها این است که روش‌های درمانی جدیدتری برای بیماری‌های بی‌شمار جامعه بشری کشف کنند. انگیزه اصلی این شرکتها برای صرف هزینه‌های سنگین تحقیقاتی این است که کلیه حقوق اکتشافات آنها، در صورتی که مورد قبول اداره ثبت اختراع واقع شود، به مدت ۲۰ سال در انحصار آنها خواهد بود. در مواردی نیز می‌توان آن را تمدید کرد.

شکل ۱-۱۰ مدل اقتصادی بازگشت سرمایه برای داروهای کلینیکی

در نقطه‌ای از چرخه تحقیق، داروی جدید کشف می‌شود. شرکت برای جلب حمایت از این کشف درخواست ثبت اختراع را می‌کند. حدود ۲ تا ۴ سال طول می‌کشد تا حق ثبت به آنها داده شود. مرحله بعد این است که کشف جدید وارد فرآیندهای توسعه شود تا قابل عرضه به بازار گردد. فرآیندهای توسعه شامل تصفیه دارو، آزمون آن بر روی موجودات غیرانسانی و ... می‌شود.

چرخه توسعه با آزمایش‌های کلینیکی تکامل می‌یابد. در این مرحله دارو به صورت کنترل شده بر روی انسان‌ها آزمون می‌شود. در این تست‌ها سلامت، میزان تأثیرگذاری و دوز مناسب آن مشخص می‌شود. اگر از سلامت دارو اطمینان حاصل گردد، چرخه تولید آن قبل از گرفتن تأییدیه از سازمانهای ذیربط، آغاز می‌شود. اما قبل از تکمیل آزمایش‌های کلینیکی و تأییدیه سازمانهای ذیربط، این دارو قابل عرضه در بازار نیست. در برخی موارد ممکن است دارو تأیید نشود که در این صورت باید فرآیند تولید متوقف گردد. در نظر داشته باشید که به محض رسیدن دارو به بازار شمارش معکوس برای انقضای حق ثبت آغاز می‌شود. شرکت تولیدکننده دارو تلاش می‌کند که توسعه و آزمایش محصول و گرفتن تأییدیه آن را در کمترین زمان ممکن انجام دهد و در عین حال محصولی سالم عرضه کند.

این شرکتها قبل از گرفتن تأییدیه اقدامات دیگری نیز انجام می‌دهند از قبیل آماده کردن تیم تولید، آموزش تیم فروش و آماده کردن بازار برای محصول فوق العاده و جدید خود. سپس چرخه عملیاتی آغاز می‌گردد. در این چرخه هدف این است که درآمد ناشی از این محصول به حداقل برسد. این محصول به تنها یعنی فقط هزینه‌های تولید، فروش، بازاریابی و توزیع بلکه هزینه سنگین چرخه‌های تحقیق و توسعه را نیز باید پوشش دهد. اگر درآمد فروش دارو به نقطه‌ای برسد که بتواند هزینه‌های عملیاتی و تحقیق و توسعه را پوشش دهد، بازگشت سرمایه آن مثبت می‌شود.

یکی از پیچیدگیهایی که در این نمودار نمایش داده نشده است، این است که در هر زمانی ممکن است تحقیقاتی بر روی محصول دیگری در حال انجام باشد و برخی از این تحقیقات ممکن است هیچ‌گاه نتیجه ندهد. آمار نشان می‌دهد که بسیاری از داروهای جدید نمی‌توانند به

بازار راه یابند بنابراین داروهایی که وارد بازار می‌شوند، باید علاوه بر پوشش هزینه‌های چرخه‌های عملیاتی و تحقیق و توسعه خود، هزینه‌های تحقیقات شکست خورده را نیز جبران کنند. از طرف دیگر درآمد این داروها باید چنان بر نقدینگی شرکت بیافزاید که علاوه بر صرف قسمتی از آن برای تحقیقات جدید، سودی نیز عاید سرمایه‌گذاران کند.

همان گونه که قبلاً گفته شد یکی از برتریهای صنعت داروسازی این است که دقیقاً می‌داند چه زمانی حق ثبت دارو منقضی می‌شود. بنابراین شرکتها می‌دانند که چه زمانی ارزش محصولشان به واسطهٔ آمدن داروهای ژنریک کاهش می‌یابد. بدین ترتیب در اینجا می‌توان مدل هزینه‌ای بسیار دقیقی طراحی کرد. به کمک آن می‌توان قیمتی برای محصول در نظر گرفت که با وجود همهٔ هزینه‌های عملیاتی، بازگشت سرمایه و سود خوبی را تضمین کند و در عین حال برای بازار قابل تحمل باشد. از آنجایی که زمان عرضه داروی ژنریک مشخص است، شرکت تولید کننده می‌تواند به موقع، اقدام به کاهش هزینه‌های عملیاتی کند. به طور کلی قابل پیش‌بینی بودن وقایع بازار طراحی مدل هزینه‌ای بسیار دقیق و مؤثری را امکان‌پذیر می‌کند، حتی اگر میزان موقوفیت تحقیقات مشخص نباشد.

۱-۲-۲- بازیابی از شرایط عمومی شدن محصول

تصور کنید چه اتفاقی می‌افتد وقتی که داروهای ژنریک در دسترس عموم قرار گیرد و بازار دیگر ارزش بالایی برای محصول شما قائل نباشد؟ در این شرایط رقابتی جدید، شرکت باید طرح تجاری جدیدی برای خود طراحی کند تا درآمد حاصل از فروش آن دارو ادامه پیدا کند. همان گونه که در نمودار می‌بینید درآمد فروش دارو به صفر نمی‌رسد. چند دلیل برای این امر وجود دارد:

- آگاهی افراد، داروخانه‌ها و پزشکان از نام تجاری (اصلی) دارو
- اعتماد به شرکت اصلی عرضه کننده دارو
- درک کردن کیفیت واقعی و تأثیر بهتر محصول اصلی
- تأثیرات سود داروهای ژنریک بر برخی افراد
- این اعتقاد در برخی افراد که باید حق الزحمه شرکتهای دارویی تا زمان استفاده از دارو پرداخت شود

این است دلیل اینکه شرکتهای داروسازی محصول خود را گران‌تر از نسخه‌های ژنریک به خوبی به فروش می‌رسانند، اما در برخی از شرایط ممکن است شرکت تصمیم بگیرد قیمت محصول خود را به قیمت داروی ژنریک نزدیک‌تر کند.

راه دیگر حفظ درآمد یک محصول این است که طرحهای تجاری جدیدتری طراحی شوند یا کاربردهای جدیدی برای محصول پیدا شود. در مورد برخی از داروها این بدین معناست که مثلاً بازار آنها به ناحیه‌های دیگر دنیا گسترش داده شود یا آنها را به طور خلاقانه‌ای بتوان ترکیب کرد تا بیماریهای دیگری را نیز بتوانند درمان کنند. بخارطه آورید بحث اول فصل را در زمینهٔ رانیتیدین و نام تجاری زانتاک. علت اینکه شما احتمالاً نام زانتاک را می‌شناسید این است که تولید کننده

دارو، «گلاکسو اسمیت کلاین» نوع با دوز کمتری از این دارو (Zantac 75) را برای فروش بدون نسخه، عرضه کرد. به کمک تبلیغات گسترده فروش زانتاک بسیار موفق بود و امروزه میلیونها نفر آن را به عنوان بهترین و مؤثرترین آنتی اسید می‌شناسند. گرچه هم اکنون انواع ژنریک رانیتیدین در بازار موجود است، اما قدرت نام تجاری و انواع بدون نسخه این دارو همچنان درآمد عاید تولیدکننده اصلی آن می‌کند.

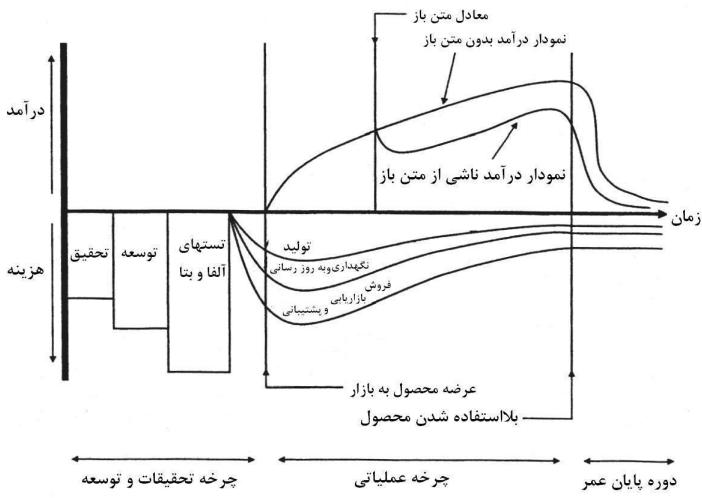
این مدل تجاری خلاقانه‌ای است که صنعت نرم‌افزار باید طراحی کند تا با آمدن محصولات متن‌باز متضرر نشود.

۱۰-۲- تأثیر متن‌باز بر نرم‌افزار

درد معده من چه ربطی به نرم‌افزار و متن‌باز دارد؟ گرچه تفاوت‌های عمدہ‌ای بین این دو وجود دارد، اما تأثیر متن‌باز بر صنعت نرم‌افزارهای تجاری بسیار شبیه تأثیر داروهای ژنریک بر صنعت داروسازی است. صنعت داروسازی سالهاست که با این پدیده دست و پنجه نرم می‌کند و یاد گرفته است که چگونه مدل‌های تجاری خود را متناسب با آن طرح‌ریزی کند. اما در صنعت نرم‌افزار، متن‌باز پدیده جدیدی است و ما هنوز در حال وفق دادن خود با تأثیرات آن هستیم. در ابتدا نگاهی می‌اندازیم به تفاوت‌های عمدہ بین این دو صنعت. یک محصول نرم‌افزاری معمولاً دارای مجموعه‌ای از امکانات و قابلیتها می‌باشد. در حالی که، یک دارو اغلب یک ترکیب شیمیایی مشخص است. ثبت اختراعی که از اجزای مشخصی تشکیل شده است، بسیار راحت‌تر از یک محصول نرم‌افزاری بزرگ است. عده زیادی در جامعه متن‌باز که مخالف ذکر حق ثبت نرم‌افزار می‌باشد. این مخالفتها از آنجا ریشه می‌گیرد که در جامعه متن‌باز مالکیت‌های معنوی آزادانه در اختیار سایر افراد داده می‌شود و آنها از بقیه همین را انتظار دارند.

یک داروی ژنریک سعی می‌کند که مشابه محصول اصلی باشد، اما در رقابت بین محصولات نرم‌افزاری حتی اگر بین محصولات متن‌باز باشد، روش‌های کاملاً متفاوتی وجود دارد. تفاوت اصلی در اینجاست که شرکتهای دارویی می‌توانند به مدت چند سال روی انحصاری بودن محصول خود حساب کنند، اما شرکتهای نرم‌افزاری به هیچ وجه نمی‌توانند انحصار دسته‌ای از محصولات را در دست گیرند.

قابلیت پیش‌بینی نیز در این دو مدل بسیار متفاوت است. صنعت داروسازی می‌تواند مدل هزینه‌ای بسیار دقیقی بر اساس عمر محصول خود بنا کند، اما در صنعت نرم‌افزار معلوم نیست چه زمان رقیبی محصولی بهتر یا ارزان‌تری عرضه خواهد کرد. اما صنعت داروسازی با همان فشارهای رقابتی سر و کار دارد که صنعت نرم‌افزار با آنها دست و پنجه نرم می‌کند و به همین دلیل منحنیهای درآمد دقیق نیستند. شکل ۲-۱۰ همان مدل قبلی است که این بار مدل‌های اقتصادی صنعت نرم‌افزار به آن اعمال شده است.



شکل ۲-۱۰ تأثیر متن باز بر ارزش محصول

جامعهٔ متن‌باز تلاش می‌کند نرم‌افزاری تولید کند که بیشترین خواهان را داشته باشد. این بدان معناست که اگر بازار محصول شما کوچک باشد، تعداد برنامه‌نویسان متن‌بازی که قصد کپی‌سازی نرم‌افزار شما را داشته باشد، بسیار کم است. اولین محصولات متن‌بازی که توسعه یافتد، کارسازهای وب، موتورهای پایگاه اطلاعات، رابطه‌ای رومیزی و ابزارهای برنامه‌نویسی بودند. تعدادی از محصولات و برنامه‌های کاربردی خاص نیز وجود دارد که هیچ معادل متن‌بازی ندارد. نکته در اینجاست که هر چه کاربرد محصول نرم‌افزاری گسترده‌تر باشد، احتمال اینکه معادل آن به صورت متن‌باز تهیه شود، بسیار بیشتر است.

البته بسیار دقّت نمائید تصور نشود صِرفِ داشتن بازار کوچک، در معرض آسیه‌های متن‌باز نیستید، بسیار احتیاط کنید. به عنوان یک مثال خوب می‌توان به ابزار پردازش و روتوش عکس و فیلم اشاره کرد. جامعهٔ متن‌باز ابزار بسیار پیشرفته‌ای برای پردازش عکس‌ها تهیه کرده است، بنام GIMP و این محصول در حال گسترش امکانات برای تدوین فیلم می‌باشد. گرچه GIMP ممکن است نیازهای همهٔ کاربران را برآورده نکند، اما معیار خوبیست از اینکه ارزش محصول شما به سرعت بخاره فشارها کاهش می‌یابد.

شکل ۲-۱۰ تأثیرات متن‌باز را بر ارزش نرم‌افزار و قدرت قیمت‌گذاری شرکتها را نشان می‌دهد. تقریباً هر نرم‌افزار تجاری دارای ارزشی خاص خود است. از دید مشتری این طبیعی است که ارزش نرم‌افزار با گذر زمان کاهش یابد. اما این کاهش ارزش نرم‌افزار، از فشارهای رقابتی و تا حدی تأثیرات خود محصول ناشی می‌شود. ۲۰ سال قبل شیشه‌های برقی برای اتومبیل گرینه بسیار گران قیمتی بود. اما امروزه اگر این امکان در اتومبیل وجود نداشته باشد، تعجب می‌کنید. فروشنده‌گان نرم‌افزار (و سازندگان اتومبیل) عموماً به این فرآیند طبیعی کاهش ارزش این گونه واکنش نشان می‌دهند که به طور پیوسته امکانات و قابلیتهای جدیدی به محصول خود اضافه

می‌کنند و محصول خود را به روز می‌کنند. این به روزرسانی محصول، ارزش آن را تجدید می‌کند. در زمینه نرم‌افزارهای شرکتی، فروشندهان این گونه نرم‌افزارها به این ترتیب می‌توانند ارزش آن را حفظ کنند که تنها مرجع قادر به پشتیبانی آن نرم‌افزار باشند. راه دیگر این است که راه ورود رقبا را بینند. وقتی که یک مشتری نرم‌افزار شما را انتخاب می‌کند، مدامی که هزینه‌های پشتیبانی و بروزرسانی آن کمتر از هزینه جایگزینی آن باشد، درآمد پیوسته‌ای عاید شما می‌شود و راه ورود رقبا بسته می‌شود.

متن باز از چندین جهت فرآیند کاهش ارزش را شتاب می‌دهد. اولاً، از آنجایی که ممکن است چندین تولیدکننده یک مشکل را هدف قرار دهند، شما زمان بسیار کمی برای تولید یک محصول قابل رقابت دارید. ثانیاً، از آنجایی که جامعه متن باز با پول تحریک نمی‌شود، همه خدماتی که شما با دریافت هزینه‌ای در اختیار قرار دهید، آنها به طور رایگان انجام می‌دهند. ثالثاً، حتی اگر به روش ذکر شده، مانع برای ورود رقبا ایجاد کنید، این مانع در مقابل متن باز بی‌تأثیر است. زیرا هزینه خرید این نرم‌افزارها اغلب صفر است. بنابراین این مانع فقط می‌تواند از جنبه پشتیبانی محصول، در مقابل متن باز باشد. به هر حال شدت و ضعف این مانع بستگی به میزان علاقه کاربر به روش‌های پشتیبانی متن باز دارد.

مشکل آخر هم قابلیت پیش‌بینی است. صنعت داروسازی می‌تواند با دقت بالا، زمان عمومی شدن محصول را پیش‌بینی کند و با مدل‌های هزینه‌ای دقیقی می‌تواند در مقابل آن واکنش نشان دهد. اما تأثیرات متن باز بسیار غیرقابل پیش‌بینی است؛ اولاً اگر فعالیتهای جاری جامعه متن باز را زیر نظر نداشته باشید، ممکن است هنگام عرضه محصول خود متوجه شوید قبل از نسخه متن باز آن آمده است. ثانیاً از آنجایی که محصولات نرم‌افزاری معمولاً ترکیبیاز امکانات و ویژگیها هستند، نمی‌توان تعیین کرد که چه زمانی یک محصول متن باز موجود، از دید مشتریان، مقبول خواهد بود.

۳-۱۰- کاهش ارزش به عنوان یک مزیت رقابتی

کلید بهره بردن از متن باز این است که یاد بگیریم چگونه از کاهش ارزش محصولات و امکانات جامعه متن باز حداکثر استفاده را ببریم، به جای اینکه حالت تدافعی نسبت به آنها اتخاذ کنیم، در اینجا تعدادی از مثالهای واکنشهای دفاعی نسبت به متن باز ذکر شده است:

- استفاده سخت‌گیرانه از حق ثبت- برخی از شرکتهای نرم‌افزاری با سلاح حق ثبت به جامعه متن باز حمله می‌کنند. این تاکتیک توصیه نمی‌شود. از آنجایی که جامعه متن باز از منابع بسیار ضعیف و محدودی برای دفاع از خود در دادگاه برخوردار است، اغلب از طریق کشاندن مسأله به رسانه‌ها و تخریب شاکی از خود دفاع می‌کند. تقریباً در اکثر موارد این امکان وجود دارد که کد برنامه چنان نوشته شود که از نظر قانونی به حق ثبت برنامه دیگری تعذر نکند و بنابراین در نهایت چیزی عاید شاکی نخواهد شد.
- شرکت Unisys انحصار الگوریتم فشرده‌سازی تصویر را که برای پرونده‌های GIF استفاده می‌شود، به نام خود ثبت کرد. در آن زمان، از این قالب گرافیکی به طور

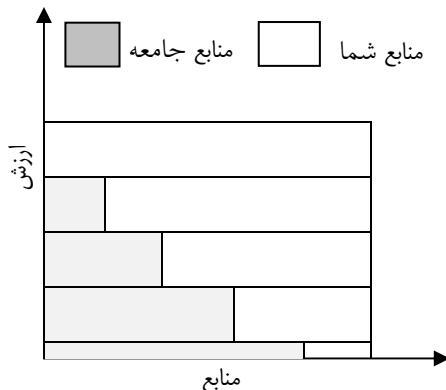
گسترهای در اینترنت استفاده می‌شد. جامعه متن باز این گونه به این اقدام پاسخ داد که از قالب‌های JPEG و PNG به جای GIF استفاده کرد.

- تهیه نرمافزار به صورت یکپارچه- امروزه اکثر نرمافزارها به صورت لایه‌های عملیاتی نوشته می‌شوند. هر لایه بر روی لایه دیگری بنا می‌شود. برنامه‌نویسان نرمافزارهای متن‌باز، نوعاً کدنویسی برنامه خود را از پائین‌ترین لایه این پُشته آغاز می‌کنند. برخی از تولیدکنندگان نرمافزار سعی می‌کنند با نوشتن برنامه به صورت یکپارچه و یکجا جلوی عمومی شدن آن را بگیرند. اما نتیجه این تلاش این می‌شود که جامعه متن‌باز نرمافزار مشابهی با معماری بهتر طراحی می‌کند و تمام امکانات نرمافزار یکپارچه قبلی را در اختیار عموم قرار می‌دهند.
- رقابت- یکی دیگر از واکنشها، تلاش برای رقابت با جامعه متن‌باز است. اما اثرات این واکنش کاملاً با گذر زمان کمرنگ می‌شود. رقابت با جامعه‌ای از برنامه‌نویسان خبره که انگیزه‌شان پول نیست و از نظر زمانی محدود نیستند، اشتباہ بزرگی است. این رقابت مستلزم صرف هزینه بسیار سنگین است و معمولاً زودتر از آنچه که تصورش را بکنید از ادامه رقابت منصرف خواهد شد.

بهترین گزینه این است که به طور فعال به ایفای نقش در این جامعه بپردازید و از کم ارزش شدن محصولات به نفع خود بهره‌برداری کنید. در فصل بعد، برخی از مدل‌های تجاری دیگر را بررسی خواهیم کرد. در اینجا فقط بر زیرساختهای فرهنگی و فکری که برای کار با این جامعه مورد نیاز است، متمرکز خواهیم شد. این فرهنگ چنان باید باشد که با دید مثبتی به جامعه متن‌باز و تحولات آن نگریسته شود.

بدون تأثیرات جامعه متن‌باز نیز محصول شما، به مرور زمان دچار کاهش ارزش خواهد شد. بنابراین شما باید همواره فناوری را چنان گسترش دهید که در قله ارزش نرمافزارها قرار داشته باشید و در این راستا مصرف منابع بالایی نیز خواهید داشت. اگر منابع خود را صرف فناوری‌کنید که نزد مشتریان از ارزش پایینی برخوردار است، سرمایه شما برنمی‌گردد.

همه شرکتهای نرمافزاری از منابع محدودی برخوردارند. در اکثر موارد شرکتها منابع‌شان را صرف بالا رفتن از قله ارزش نرمافزارها می‌کنند. اما ترفندی که باید در اینجا به کار گرفته شود این است که تلاش کنید با جامعه متن‌باز شریک شوید تا بتوانید از منابع آن بھر گیرید. سپس از آن منابع برای تولید محصول بالرتبه‌تر استفاده کنید. در شکل ۳-۱۰ نسبت استفاده از منابع جامعه متن‌باز و شرکت برای عرضه یک محصول دارای ارزش خاص، نمایش داده شده است.



شکل ۱۰-۳: نمودار رابطه منابع، با ارزش آنها

سیاست شما باید این باشد که با بالا رفتن ارزش محصول، منابع بیشتری به آن اختصاص دهید. در فصل بعدی در زمینهٔ نحوهٔ ترکیب نرم‌افزارهای اختصاصی و متن‌باز بحث خواهد شد. نکته‌ای که باید به آن دقت شود این است که حتی در پایین‌ترین سطح ارزش نرم‌افزار، باید قسمتی از منابع خود را به آن اختصاص دهیم. نباید چنین تصور کنید که می‌توان نرم‌افزار را به جامعهٔ سپرد و در عین حال بر آن نظارت داشت. باید حداقل یک نفر از شرکت شما پروژه را هدایت و فعالیت جامعهٔ متن‌باز را با اهداف شرکت همراستا کند. او همچنین باید مطمئن شود که لایهٔ نرم‌افزار در حال توسعه، نیازهای لایهٔ بالاتر را برآورده می‌کند. در نهایت باید خودتان نگهدارندهٔ آن نرم‌افزار باشید.

مزیت دیگر حفظ ارتباط نزدیک با جامعهٔ متن‌باز این است که در تمامی سطوح توسعهٔ نرم‌افزار از نحوهٔ تأثیر احتمالی جامعهٔ بر ارزش نرم‌افزار خود آگاه می‌شود و می‌توانید به موقع جهت‌گیری پروژه را تغییر دهید. بنابراین معیارهای دقیقی از نحوهٔ پیشبرد پروژه در دست خواهید داشت.

۱۰-۴- ارزش در حصار زمان

در یکی از کنفرانس‌های Linux World، من با «تیم اورایلی» (مدیر انتشارات «O'Reilly») برای صرف نوشیدنی همراه شدم. ما دربارهٔ مسائل مختلف متن‌باز صحبت کردیم و من عبارت «ارزش به عنوان تابعی از زمان» را به کار بردم و مفهوم این عبارت را برای تیم توضیح دادم. سپس او دیدگاه خودش را از مفهوم این عبارت برای من توضیح داد که با دیدگاه من متفاوت بود. نظر او این بود که جامعهٔ متن‌باز برای عرضه یک محصول مفید و کار، محدود به یک بازه زمانی خاص نیست. هر کس که با صنعت فناوری‌های پیشرفته در ارتباط باشد، به خوبی می‌داند که محصولات باید قبل از موعدشان عرضه شود. اگر این زمان بگذرد، این محصول دیگر روشنایی روز را نخواهد دید و در انبارها خاک خواهد خورد. جامعهٔ نرم‌افزارهای آزاد درگیر چنین محدودیتهاست و هر فناوری در هر زمانی استفاده خودش را پیدا خواهد کرد. پس از این ملاقات، توضیحات او فکر مرا به خود مشغول کرد و به این نتیجه رسیدم که خود من هم درگیر مسئلهٔ «ارزش در حصار زمان» بوده‌ام. در ادامه تجربه خود را در این زمینه بازگو خواهم کرد.

۵-۱۰- کلاس‌های بنیادی Pilot

در سال ۱۹۹۶، شرکت US Robotics یک PDA^۱ (دستگاه ذخیره اطلاعات دستی) استثنائی به نام Pilot 5000 روانه بازار کرد. امروزه ما تکامل یافته همان محصول را با نام Palm می‌شناسیم. زمانی که این محصول عرضه شد من بالاصله یکی از آن را تهیه کردم و تصمیم گرفتم آن را برنامه‌ریزی کنم. سامانه‌عامل Pilot (که امروزه Palm OS نامیده می‌شود) بر اساس مجموعه توابعی به زبان C بنا شده است. من با برنامه‌نویسی شیء^۲ گرا آشنا بودم و قصد داشتم این دستگاه را با زبان C++ برنامه‌ریزی کنم. این کار از نظر فنی امکانپذیر بود، اما وقتی عملی می‌شد که من می‌توانستم از توابع Palm OS به عنوان مجموعه‌ای از کلاس‌های C++ در برنامه‌های خود استفاده کنم، من باید تقریباً همان کاری را می‌کردم که میکروسافت برای ساختن کلاس‌های بنیادی خود (MFC) انجام داده بود. مجموعه‌ای از کتابخانه‌های است که برنامه‌نویسی ویندوز را بسیار آسانتر می‌کند. به این ترتیب که توابع Win32 را به صورت کلاس‌هایی در اختیار برنامه‌نویسان قرار می‌دهد.

در نهایت خودم این مجموعه کلاسها را برای Palm OS نوشتیم و آن را کلاس‌های بنیادی Pilot نامیدم. من این پروژه را به صورت پاره‌وقت، در حین سفرهایم به آن سوی اقیانوس اطلس، انجام می‌دادم. گرچه من این پروژه را تکمیل کرده بودم اما برنامه‌نویس دیگری نیز پروژه مشابهی را تکمیل و عرضه کرد. من دلیلی نمی‌بینم که بخواهم با او رقابت کنم. من به اندازه کافی بستر OS Palm را شناخته بودم و از وقتی که صرف آن کرده بودم، راضی بودم. حداقل چیزی که برای من ماند تجربه‌ای بود که از نوشتن کلاس‌های بنیادی، بدست آوردم.

۶-۱۰- Jump

یکی از مهم‌ترین کسانی که به محیط Palm خدمت کرد، گرگ هیوجیل^۳ بود. گرگ یک شبیه‌ساز بسیار جالب با نام Co Pilot عرضه کرده بود که نقطه اتکاء هر برنامه‌نویسی بود (در فصل بعد این شبیه‌ساز را بررسی خواهیم کرد). حدود سال ۱۹۹۷ گرگ پروژه دیگری را به نام Jump آغاز کرد. اگرچه در تعریف رسمی Jump گفته شده است که از ابتدای کلمات Java User Module گرفته شده است، اما گرگ ادعا می‌کند که این نام را از آهنگی که موقع هک کردن گوش می‌داد، گرفته است. Jump به برنامه‌نویسان این امکان را می‌داد که برنامه‌های خود را به زبان جاوا بنویسند و سپس به کمک آن برنامه خود را به کد ماشین دستگاه‌های Palm تبدیل کنند. آنها که با زبان جاوا آشنا‌یابی دارند می‌دانند که برنامه‌های جاوا اغلب توسط یک ماشین مجازی به نام JVM (ماشین مجازی جاوا) اجرا می‌شوند. اما Jump برنامه‌های جاوا را مستقیماً تبدیل به کد ماشین پردازنده موتورولا k 68 می‌کند. وقتی که گرگ در حال آماده کردن نسخه بتای Jump بود، من روی برنامه نمایشی Word Clock کار می‌کردم که بعداً نام آن را به Desk

¹ Personal Digital Assistant

² Greg Hewgill

Clock تغییر دادم (در شکل ۴-۱۰ نمونه‌ای از خروجی این برنامه‌ها را می‌بینید). این برنامه ساعت سه منطقه زمانی را به طور همزمان بر روی صفحه نمایش نشان می‌داد. گرگ قصد داشت برنامه‌های نمایشی مرا به عنوان برنامه‌های کمک آموزشی به همراه Jump ارائه کند.



شکل ۴-۱۰: نمونه‌ای شبیه ساز DeskClock در حال اجرای PalmOS

اینجا بود که من به مهارت کسب شده در طراحی کلاس‌های بنیادی مراجعه کردم. Java زبانی شی‌گرایست و طراحی کلاس‌های بنیادی Pilot JFC برای توابع Java هدف بعدی من بود. این بار ما چند نفر بودیم که بر روی Pilot JFC کار می‌کردیم. اما این‌بار بیشتر از منابع جامع متن باز استفاده کردیم، این است نمونه‌ای از فناوری‌ای که من طراحی کردم و چنان مورد استفاده برنامه‌نویسان قرار گرفت که فکرش را هم نمی‌کردم.

گرچه به دلایل نامشخصی، گرگ کار بر روی Jump را به طور ناگهانی رها کرد. Jump محدودیتهایی داشت که استفاده از آن را به عنوان یک بستر برنامه‌نویسی مشکل می‌ساخت. به تدریج جامعه‌ای که روی Jump کار می‌کرد از هم پاشید. گرگ متلاعده شد از آنجایی که دیگر روی Jump کار نمی‌کند، باید که منبع آن را منتشر کند تا بقیه و افراد دیگری کار او را ادامه دهند. او Jump را تحت مجوز GPL عرضه کرد. من هم دیگر سیر تحولات Jump را پیگیری نکرم. چند سال قبل برنامه‌نویس دیگری به نام «رالف کلبرف» پروژه Jump را با نام Jump2 تکمیل کرد.

تصور کنید من چقدر هیجان‌زده شدم وقتی که بعد از ۴ سال، ارتباط نداشتن با برنامه‌نویسی، پست الکترونیک زیر را از جیم برچیلید دریافت کردم. او این ایمیل را برای برنامه‌نویس JFC دیگری نیز فرستاده بود.

از: جیم برچفیلد

تاریخ ارسال: سه شنبه ۲۷ فوریه ۲۰۰۱، ساعت: ۱۳:۳۵ بعدازظهر

به: مارتین فینک

موضوع: احیاء JFC

من اخیراً شروع به کار با Jump کردهام و متوجه شدم که در برخی از اظهارنظرهای قدیمی، به Pilot JFC اشاره شده است.

من نسخه قبلی JFC را ارتقاء داده‌ام و آن را با آخرین نسخه Jump منطبق کرده‌ام. من Word Clock را نیز به این معماری جدید منتقل کرده‌ام. در حین این کار برخی از امکانات آن را بهبود داده‌ام. البته باید بگوییم که آخرین نسخه آن نیز به خوبی کار می‌کند.

آیا هیچ یک از شما می‌توانید مرا در جریان وضعیت کنونی پروژه JFC قرار دهید؟ من قصد دارم این پروژه را به صورت امکانات اضافه توزیع Jump عرضه کنم. همچنین مایلم توسعه و نگهداری آن را نیز بر عهده گیرم. اما دوست ندارم بینم نسخه‌های مختلف و ناهمنگی از آن عرضه شده است.

بنابراین من می‌خواهم کد منبع آن را در اختیار داشته باشم، با این شرط که همه جا نام شما را ذکر کنم. همچنین تعدادی برنامه نمایشی خواهم نوشт و آنها را از اجزاء اصلی Pilot JFC قرار خواهم داد.

تیم اورایلی باعث شد که من متوجه موضوعی شوم که خودم قبل آن را تجربه کرده بودم. نرمافزاری که توسط چندین نفر نوشته شده بود، آنقدر خاک خورد تا اینکه نیاز به آن از طرف جامعه کاربران احساس شد.

این مثال باید برای همکاری بیشتر با جامعه متن‌باز انگیزه بیشتری در شما ایجاد کند. حتی اگر پروژه‌های برنامه‌نویسی شما متوقف شده‌اند یا با سرعت بسیار کمی پیش می‌روند، آنها را دست کم نگیرید. تنها چیزی که آنها احتیاج دارند یک برنامه‌نویس با انگیزه است تا فعالیت روی آن را از سر گیرد و آن را به عنوان یک محصول متن‌باز در اختیار عموم قرار دهد. شاید روزی خود شما تبدیل به آن فرد با انگیزه شوید و اجرای پروژه را سرعت بخشید. به عنوان حسن ختام به چند نکته در زمینه پستالکترونیک جیم اشاره می‌کنم. اولاً او قبل از بدست گرفتن کاری از نویسنده‌گان اصلی آن اجازه گرفت. ثانیاً او به من اطمینان داد که نام مرا در کارهایش ذکر می‌کند. ثالثاً دقت کنید که او از کلمه توزیع برای Jump به همان مفهومی که من در توزیعهای غیرلینوکسی در فصل ۵ توضیح دادم، استفاده کرده است. نهایتاً اینکه او قصد ندارد که با سایر تلاشهای انجام شده در جامعه رقابت کند. همین پستالکترونیک کوتاه توانسته است نمایانگر فرهنگ جامعه متن‌باز باشد.

۷-۱۰- جمع‌بندی

در این فصل تاثیرات اقتصادی نرم‌افزارهای متن‌باز بر محصولات شما شرح داده شد. در پاسخ به تاثیرات آنها شما می‌توانید به آنها حمله کنید، دفاع کنید یا حتی با آنها رقابت کنید. اما بهترین را این است که با این جامعه همکاری و شراکت کنید و با استفاده از شرایط به وجود آمده، منابع خود را بر فعالیتهایی متتمرکز کنید که منفعت بیشتری برای شما داشته باشد. مدیران فناوری اطلاعات نیز با استفاده از شرایط بوجود آمده باید بسرعت پروژه‌های خود بیفزایند. همچنین دیدید که همکاری کردن با جامعه باعث می‌شود با پروژه‌های در حال انجام و حتی متوقف شده آشنا شوید. بدین ترتیب می‌توانید مسیر تکامل مدل تجاری شرکت خود را بهتر ترسیم کنید. حال که آثار اقتصادی متن‌باز و بهترین پاسخ نسبت به آن را یاد گرفتیم، زمان آن رسیده است که نحوه تکامل مدل تجاری شرکت را بررسی کنیم. در فصل بعد جزئیات مدل‌های تجاری که می‌توانند از متن‌باز بیشترین بهره را ببرند، بررسی خواهد شد.

فصل یازدهم

مدلهای تجاری درآمدزا

به عنوان یک تولیدکننده محصولات نرمافزاری، بیشک زمانی فرا خواهد رسید که متوجه خواهید شد لینوکس و متن باز تأثیر مستقیمی بر جریان درآمد شما می‌گذارد. در این شرایط باید نحوه توسعه محصولات را اصلاح کنید و به منابع درآمدی خود از جنبه دیگری نگاه کنید. همچنان که با لینوکس، متن باز و مفهوم نرمافزار آزاد آشنایی شوید، باید نحوه درآمدزایی را با تولید محصولات، خدمات و پشتیبانیهای جدید یاد بگیرید. در این فصل چند مدل تجاری بررسی خواهد شد. از آنجایی که کسب و کارها متفاوت هستند، هر یک از این مدلها را باید با شرایط کسب و کار خود سازگار کنید. هدف این فصل این است که چندین مدل تجاری و نحوه استفاده از آنها در کسب و کار خود را به شما بیاموزد.

این فصل نیز برای شرکتهای تولیدکننده محصولات نرمافزاری نوشته شده است، اما همچنان برای مدیران فناوری اطلاعات مفید است. تحول در مدل تجاری تولیدکنندگان نرمافزار بدون شک بر نحوه خرید مدیران فناوری اطلاعات نیز تأثیر می‌گذارد. مدلها تجاری که در یک فصل بررسی خواهند شد عبارتند از:

- نرمافزار تجاری برای لینوکس
- خدمات و پشتیبانی یک محصول متن باز
- ارتقاء و بهبود محصولات تجاری به کمک نرمافزارهای متن باز
- نرمافزار متن باز برای فعال سازی اجزای سخت افزاری
- تجاری کردن یک محصول متن باز از طریق مجوز دوگانه
- کاهش هزینه پایان چرخه زندگی
- ایجاد یک محیط

جنیش نرمافزار آزاد و اکثر فعالان جامعه متن باز معتقدند که نرمافزار باید آزاد باشد. گرچه منظور از آزادی در دسترس بودن کد منبع است، اما در اغلب موارد به مفهوم رایگان نیز می‌باشد. در این فصل قصد نداریم بگوییم نرمافزارهای شما باید رایگان باشند. متن باز در صورتی که نحوه پیاده سازی آن را بلد باشید، می‌تواند در فرآیند بسیار مفید واقع شود.

این فصل جزئیات مدلها تجاری مختلف را بررسی خواهد کرد. در فصل بعد فرآیند استفاده از متن باز در محصولات داخلی و عرضه یک نرمافزار متن باز را به جامعه بررسی خواهیم کرد.

۱۱-۱- ارزش محصول خود را بدانید

اولین قدمی که باید بردارید این است که ماهیت کسب و کار خود را روشن کنید. مشتریان شما چه کسانی هستند و چه چیز با ارزشی را به آنها عرضه می‌کنند؟ در فصل گذشته یاد گرفتید که هر محصول با ارزشی با گذر زمان عمومی خواهد شد و از انحصار شما درخواهد آمد. متن باز به فرآیند عمومی شدن شتاب بیشتری می‌دهد. بنابراین هدف شما باید این باشد که همواره

محصول بالارزش‌تری به مشتریان خود عرضه کنید. این بدین معناست که اگر محصول شما در حد بهترین محصول موجود نباشد، مشتریان آن را نمی‌خرند.

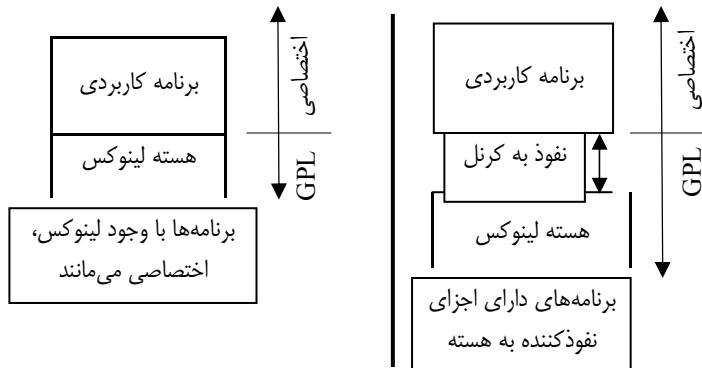
اگر از جایگاه خود نزد مشتریان آگاهی داشته باشید به راحتی می‌توانید آنچه را که مورد نیاز کسب و کارتان است، تشخیص دهید و بخشهایی را که دیگر نمی‌تواند برای شما پول‌ساز باشند عمومی کنید. متن‌باز از این طریق می‌تواند تأثیر مثبتی بر کسب و کار شما داشته باشد. با بررسی و ارزیابی دقیق جامعه متن‌باز، شما می‌توانید همواره روی بالارزش‌ترین محصول سرمایه‌گذاری کنید. این فرآیند می‌تواند مدیران شما را دستپاچه کند. آنها از این پس باید منابع تولید محصول خود را از جاهای دیگری تأمین کنند. اصلاح فرهنگی که باید در سازمان صورت دهد، این است که مدیران و کارمندانشان به طور پیوسته در فکر تولید محصولات بالارزش‌تر باشند و در عین حال همکاری با جامعه متن‌باز نیز تشویق شود.

با فرض اینکه شما سیاستهای تجاری سازمان و اصل ارتقاء ارزش محصول را به درستی درک کرده‌اید، چند مدل تجاری برای پیاده کردن در سازمان در اختیار خواهید داشت.

۱۱-۲- نرم‌افزار تجاری و لینوکس

این ساده‌ترین مدل تجاری مطرح شده در این فصل است. با این حال باید به چند نکته اشاره شود. بسیاری افراد تصور می‌کنند که نمی‌توان نرم‌افزارهای تجاری و اختصاصی را تحت لینوکس اجرا کرد. این تصور غلط است. به غیر از چند استثناء، نرم‌افزارهای کاربردی اختصاصی تحت لینوکس را همانند نرم‌افزارهای اختصاصی سایر سامانه‌های عامل، می‌توان به فروش رساند. امروزه به نمونه‌های زیادی از نرم‌افزارهای تجاری تحت لینوکس می‌توان اشاره کرد. در مجوز لینوکس (GPL) هیچ الزامي برای انتشار کد منبع این گونه نرم‌افزارها نشده است. در شکل ۱۱-۱ شما از نحوه اجرای نرم‌افزارهای اختصاصی و نرم‌افزارهای نفوذ‌کننده به هسته در لینوکس دیده می‌شود.

برخی از نرم‌افزارهای تجاری از مرزهای هسته لینوکس فراتر می‌روند. در این گونه نرم‌افزارهای خاص باید فرآیند تولید و یکپارچه‌سازی نرم‌افزار به صورت دیگری انجام شود. در اینجا باید چند تصمیم‌گیری تجاری انجام دهید. از آنجایی که هسته لینوکس به صورت GPL مجوزدهی شده است، قسمتهایی از نرم‌افزار شما که در هسته انجام می‌شود باید تحت GPL مجوزدهی شود. توصیه می‌کنم فصلهای ۲ و ۳ را، برای تصمیم‌گیری در زمینه پیمانه‌های هسته و مجوز GPL مرور کنید.



شکل ۱-۱۱: اجرای نرم‌افزارهای تجاری در لینوکس

اگر بدانید که از دید مشتری ارزش محصول شما کجاست، متوجه می‌شوید که اجزاء نفوذ‌کننده در هسته برای مشتریان شما ارزشی ندارند. بنابراین این اجزاء را می‌توانید به صورت GPL عرضه کنید. اما اگر کسب و کار شما به هر دلیلی اجازه ندهد که هیچ قسمتی از نرم‌افزار نفوذ‌کننده به هسته GPL شود، نمی‌توانید آن را برای لینوکس عرضه کنید. در فصل ۲ آموختید که می‌توان پیمانه‌های اختصاصی برای لینوکس تهیه کرد. اما این روش توصیه نمی‌شود، بنابراین آن را بررسی نمی‌کنیم. اگر مز بین بخش‌های نفوذ‌کننده به هسته و سایر بخش‌های نرم‌افزار مشخص شده باشند، بدون هیچ مشکلی می‌توانید نرم‌افزار خود را عرضه کنید. بخش‌های نفوذ‌کننده به هسته، به صورت پیمانه‌ها یا وصله‌هایی به هسته لینوکس عرضه می‌شود. سپس می‌توانید قسمتهای اختصاصی نرم‌افزار خود را به این پیمانه‌ها متصل کنید و محصول را به مشتری خود عرضه کنید.

۱۱-۳- پشتیبانی و خدمات نرم‌افزارهای متن‌باز

توزیع کنندگان لینوکس یکی از اولین مدل‌های تجاری در جامعه متن‌باز را طراحی کرده‌اند. ابتدا بین گونه این مدل تجاری این است که توزیع کنندگان لینوکس، پشتیبانی توزیع لینوکس خود را به مشتریان می‌فروشند. از آنجایی که توزیع کنندگان لینوکس توزیع خود را با اجزاء متن‌باز دیگر که خودشان طراحی کرده‌اند عرضه می‌کنند، در زمینه یکپارچه‌سازی تبحر زیادی دارند. این توزیعهای متن‌باز چندین مدل تجاری را ایجاد کرده‌اند.

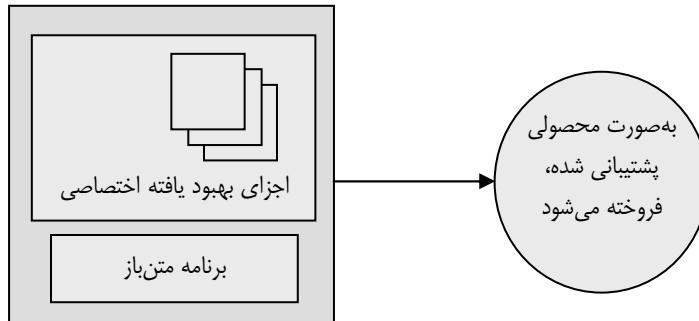
- تهیه بسته‌های نرم‌افزاری برای فروش - بسیاری از مشتریان ترجیح می‌دهند رسانه حاوی نرم‌افزار و راهنمایی آن را با پرداخت هزینه‌ای دریافت کنند. آنها نمی‌دانند که یک کپی از آن نرم‌افزار را می‌توان کاملاً قانونی و رایگان از طریق اینترنت دریافت کنند. بنابراین بسیاری از توزیع کنندگان لینوکس توزیع خود را به صورت بسته‌های نرم‌افزاری برای فروش عرضه می‌کنند.

- پشتیبانی - توزیع کنندگان لینوکس اغلب خدمات پشتیبانی خود را نیز به همراه بسته نرم‌افزاری عرضه می‌کنند تا بتوانند مشتری را در حل مشکلات پیش آمده کمک کنند. مشتریان شرکتی بزرگ خدمات پشتیبانی گسترده‌تری را برای لینوکسهای خود درخواست می‌کنند. همچنین برخی از شرکتهایی که توزیع کننده لینوکس نیستند، مانند Linux Care، خدمات پشتیبانی برای طیف گسترده‌ای از محصولات متن‌باز ارائه می‌کنند.
- آبونمان - آبونمان بخشی از مجموعه خدماتی است که توسط برخی از توزیع کنندگان عرضه می‌شود. خدمات آبونمان امکاناتی را در اختیار مشتری قرار می‌دهد که همواره نرم‌افزار خود را به روز نگه دارند. شرکت زیمیان (Ximian) از خدمات آبونمان برای به روز کردن محیط رومیزی گنوم استفاده می‌کند.
- نام تجاری - برخی از فروشنده‌گان نرم‌افزار و سخت‌افزار از توزیع لینوکس خاصی استفاده می‌کنند و آن را با نام تجاری خود مرتبط می‌کنند و در فعالیتها و تبلیغات نام آن را ذکر می‌کنند. بستگی به شرایط، فروشنده ممکن است برای این کار هزینه‌ای نیز دریافت کند. این مرتبطسازی باعث کسب اعتبار بیشتر برای آن توزیع یا فروشنده می‌شود و در جذب مشتریان مؤثر است.
- خدمات حرفه‌ای - کار پیوسته روی یک توزیع لینوکس، قدرت رقابتی توزیع کننده را بهتر نمایش می‌دهد. مثلاً، شرکت Red Hat در زمینه ارائه خدمات حرفه‌ای برای لینوکس خود تخصص دارد.

برای اینکه بتوانید مدل‌های تجاری فوق را پیاده‌سازی کنید، ابتدا لازم است کسب و کار خود را به طور دقیق تحلیل کنید و به کمک آن هزینه‌های کسب اعتبار و درآمد ناشی از آن را تخمین زنید. برخی از شرکتها به این نتیجه رسیده‌اند که ادامه دادن تجارت با این روش مشکل است و در حال مدل تجاری کنونی خود (مبنی بر متن‌باز) با مدل‌های سنتی (مثلاً فروش نرم‌افزارهای اختصاصی) هستند.

۱۱-۴- ترکیب و بالا بردن ارزش

در این مدل تجاری نرم‌افزارهای متن‌باز با نرم‌افزارهای اختصاصی به گونه‌ای ترکیب می‌شوند که راه حل‌های بهتری در اختیار مشتریان قرار دهند. در برخی موارد نرم‌افزارهای متن‌باز با یک محصول سخت‌افزاری ترکیب می‌شوند (در زمینه مدل تجاری سخت‌افزار، در همین فصل توضیح خواهم داد). اما در اکثر موارد شما می‌توانید نرم‌افزار اختصاصی خود را به کمک اجزای متن‌باز بهبود دهید. راهکار دیگر این است که امکانات یک نرم‌افزار اختصاصی را به نرم‌افزاری متن‌باز اضافه کنید و بسته نرم‌افزاری حاصل را بفروشید. شکل ۲-۱۱ نحوه ترکیب یک نرم‌افزار متن‌باز با اجزاء اختصاصی را نشان می‌دهد.



شکل ۱۱-۲: بهبود یک نرمافزار متن باز به کمک اجزای اختصاصی

به عنوان یک مثال خوب از این مدل تجاری می‌توان به محصولی از شرکت فناوری «کوالنت» اشاره کرد. کارساز وب آپاچی پر استفاده‌ترین کارساز وب در اینترنت است و به نرمافزار متن باز گشته معروف است. کوالنت امکانات مورد نیاز شرکتهای بزرگ را به این کارساز می‌افزاید. این شرکت امکانات مدیریتی، امنیتی و غیره را به این کارساز می‌افزاید و آن را آماده استفاده تجاری می‌کند. از محصولات دیگر کوالنت، خدمات پشتیبانی برای شرکتهای بزرگ است. از آنجایی که تیم برنامه‌نویسان این شرکت، ارتباط نزدیکی با جامعه متن باز و شرکتهای تجاری دارند، پل ارتباطی مناسبی برای ارتباط مشتریان با این جامعه هستند.

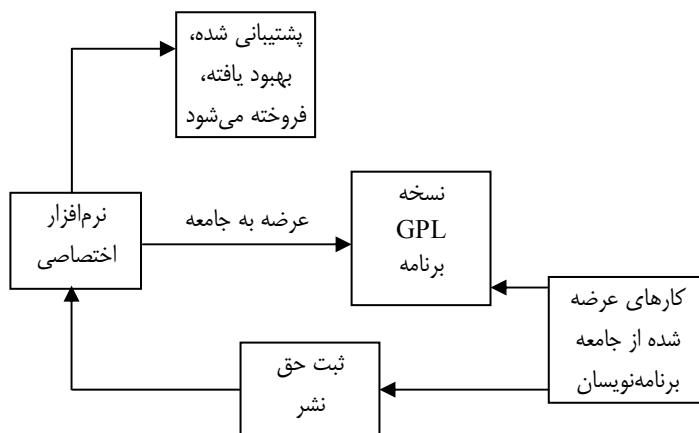
برای هر شرکتی این امکان وجود دارد که امکانات تجاری به کارساز آپاچی اضافه کند و حتی به انجام این کار با سایر شرکتها رقابت کند. اگر این کار درست انجام شود، در حالی که شرکتها برای عرضه کارایی با ارزش بالاتر رقابت می‌کنند، منجر به بهبود پیوسته این کارساز نیز خواهد شد. این شرکتها می‌دانند ارزشی که به آپاچی افزوده شده است، ناشی از ابزاریست که آنها طراحی کرده‌اند.

این مدل را می‌توان به هر یک از بسترها اصلی متن باز اعمال کرد مانند کارسازهای اینترنتی، فناوری‌های خوشه‌سازی (کلاسترینگ)، سامانه‌های پرونده، رابطه‌ای کاربری، ابزار گرافیکی و غیره. بدین ترتیب شما می‌توانید بدون صرف انرژی بر روی اجزای سطح پایین که قبلاً طراحی شده‌اند، محصولات بالرزشی تولید کنید.

۱۱-۵- تجاری‌سازی با یک مجوز دوگانه

مفهوم مجوز دوگانه را در فصل سوم به طور مختصر شرح دادم و سامانه‌پرونده FS Reiser را به عنوان مثال آوردم. در این قسمت مجوز دوگانه به عنوان یک مدل تجاری بررسی خواهد شد. در این مدل خاص نیازمند در اختیار داشتن مجوز کپی، یا هر مجوز قابل قبول دیگری برای تمامی کدهای منبع هستیم. مجوز دوگانه بدین معناست که شما یک نسخه متن باز از نرمافزار را نگهداری کنید و همزمان یک نسخه از همان نرمافزار را تحت مجوزهای تجاری عرضه کنید.

این مدل تجاری حول مجوزهای سنتی نرمافزار و درآمد ناشی از پشتیبانی نسخه تجاری، می‌چرخد. این مدل اغلب تفاوت‌هایی بین نسخه‌های متن‌باز و تجاری نرمافزار قائل می‌شود. این تفاوتها اغلب به این صورت هستند که نسخه‌های جدید نرمافزارها که به صورت تجاری عرضه می‌شوند، کاملاً آزمون شده‌اند و اشکالات کمتری دارند و ممکن است امکاناتی در آنها باشد که در نسخه متن‌باز نباشد. در این مدل یک پارامتر زمان هم وجود دارد. بدین معنا که نسخه‌های متن‌باز چندین ماه بعد از نسخه‌های تجاری عرضه می‌شوند. شکل ۳-۱۱ نمای کاملی است از این مدل.



شکل ۳-۱۱: مدیریت نرمافزار تجاری دارای مجوز دوگانه

در میان مزایای این مدل تجاری می‌توان به امکان بهره‌گیری از برنامه‌نویسانی که به نرمافزار شما علاقمند هستند، اشاره کرد. شما می‌توانید از این فرصت برای کاهش هزینه‌های برنامه‌نویسی استفاده کنید. نسخه متن‌باز نرمافزار باعث ایجاد جمعیت بزرگ در کاربران علاقمند به این نرمافزار می‌شود و این خود تبلیغی است برای محصول شما. گرچه ممکن است نسخه تجاری نرمافزار از نسخه متن‌باز بهتر باشد، اما جامعه متن‌باز را ارضاء می‌کند. هر چند در دسترس بودن نسخه متن‌باز باعث از دست دادن بخشی از درآمد این نرمافزار می‌شود، اما گسترش کاربران و آگاهی جامعه از امکانات این محصول، اغلب منجر به افزایش تعداد مشتریان شما می‌شود.

برای پیاده‌سازی این مدل باید تمامی یا بخشی از نرمافزار خود را تحت یکی از مجوزهای متن‌باز عرضه کنید. در فصل بعد شرایط عرضه محصول بررسی خواهد شد که یکی از آنها مالکیت حق کپی تمامی قسمتهای نرمافزار می‌باشد. شرکت شما باید وظیفه نگهداری نسخه متن‌باز نرمافزار را نیز به عهده گیرد. به عنوان نگهدارنده نرمافزار، پذیرش یا رد اصلاحات عرضه شده بر عهده شماست. پس از تصمیم‌گیری در این زمینه، باید از اشخاصی که اصلاحات قابل

قبولی ارائه کرده‌اند، درخواست کنید که حق کپی آن را به شما واگذار نمایند. بنابراین حق کپی کارهای عرضه شده توسط دیگران نیز در اختیار شرکت شما خواهد بود. در ضمیمه «ب» یک نمونه از نحوه واگذاری حق کپی آمده است. به هر حال در تمامی مراحل باید از مشاوره یک حقوقدان استفاده کنید تا فعالیتهای شما برخلاف قانونهای محلی و جهانی نباشد. در واگذاری حق کپی اغلب این امکان به صاحب آن داده می‌شود که یک نرمافزار را هم تحت مجوزهای اختصاصی و هم تحت مجوزهای متن‌باز عرضه کند. اگر حق کپی نرمافزاری در انحصار شما باشد، می‌توانید آن را با هر نرمافزار اختصاصی دیگری که مالکیت آن را در اختیار دارید، ترکیب کنید.

۱۱-۶- سخت افزار

یکی از مدل‌هایی که به آسانی می‌توان پیاده‌سازی کرد، مدل تجاری سخت افزار می‌باشد. این مدل فرض می‌کند که ارزشی را که به مشتری ارائه می‌کنید، در سخت افزار است. سخت افزاری که عرضه می‌کنید، معمولاً نوعی دستگاه جانبی یا سخت افزاری رابط برای یک دستگاه جانبی است. به عنوان مثالهایی از دستگاه جانبی می‌توان به پرینتر، اسکنر، دیسک‌گردان اشاره کرد. کارت‌های گرافیکی، کنترلرهای دیسک و رابطهای USB نمونه‌هایی از رابطهای سخت افزاری هستند.

اگر هدف اولیه شما عرضه سخت افزار است باید کاربرد آن را به جاهای بیشتری گسترش دهید. یکی از راه‌های گسترش بازار با کمترین هزینه این است که از جامعه متن‌باز برای نوشتمن راه‌اندازی سخت افزار خود (نرم افزار رابط بین سامانه‌عامل و سخت افزار) بهره گیرید.

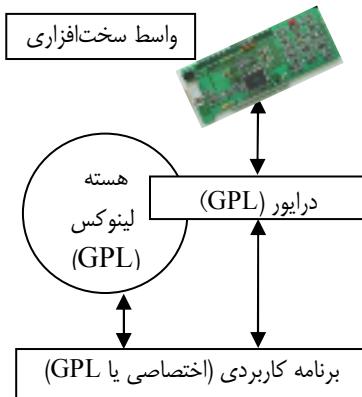
۱۱-۶-۱- تمایز بین دستگاه واسط و کارآئیهای آن

برای توضیح این مفاهیم، مثالی می‌زنم. فرض کنید شما رابط سخت افزاری می‌فروشید که امکان تماشای تلویزیون بر روی رایانه را در اختیار مشتریان می‌گذارد. سخت افزار شما نوعی کارت واسط خواهد بود که داخل رایانه نصب می‌شود و می‌تواند سیگنال تلویزیون را دریافت کند. در این مثال حداقل دو نرم افزار باید همراه این دستگاه عرضه شود. یکی نرم افزار راه‌انداز است که کارت واسط را کنترل می‌کند و دیگری نرم افزاری است برای نمایش تلویزیون. سؤالات مدل تجاری شما باید اینها باشند:

- آیا مشتری من ارزش نرم افزار راه‌انداز را می‌داند؟

- آیا مشتری من ارزش نرم افزار نمایشگر تلویزیون را می‌داند؟

برای اینکه بتوانید از جامعه متن‌باز بهره گیرید و بازار خود را گسترش دهید باید به سؤال اول پاسخ منفی دهید. پاسخ سؤال دوم ممکن است منفی یا مثبت باشد. در شکل ۱۱-۴ ارتباط بین نرم افزار، راه‌انداز و سخت افزار واسط نمایش داده شده است.



شکل ۱۱-۴: واسطه‌های سختافزاری

با دادن پاسخ منفی به سؤال اول، شما تضمین می‌کنید که این سختافزار در بسترها مختلف قابل استفاده است و این موقعیت شما را در بازار بهبود می‌بخشد. بنابراین قدم بعدی شما باید این باشد که ارتباط نزدیکی را با جامعه متن باز آغاز کنید و آنها را تشویق به پیاده‌سازی سختافزار شما در بسترها سختافزاری و نرمافزاری مختلف کنید. همان‌گونه که در فصل قبل اشاره شد باید حداقل منابعی را به این پروژه اختصاص دهید تا بتوانید آن را در جامعه متن باز هدایت کنید. اگر با جامعه متن باز همکاری کنید آنها اکثر کار برنامه‌نویسی را برای شما انجام خواهند داد. بدین ترتیب می‌توانید چرخه توسعه را سریعتر کنید و مثلاً با دادن سختافزار تولیدی خود به اعضای کلیدی این جامعه، ارتباط خود را با آنها مستحکم کنید. اگر هزینه تولید سختافزار شما چند صد دلار باشد، معامله بسیار خوبی انجام داده‌اید. در زمینه میزان کنترلی که می‌خواهید بر روی نرمافزار راهاندازها داشته باشید نیز باید از قبل تصمیم گرفته باشید. این تصمیم بستگی به این دارد که چگونه می‌خواهید این نرمافزار را پشتیبانی و نگهداری کنید. ممکن است توسعه نرمافزار راهانداز را به یکی از برنامه‌نویسان کلیدی جامعه متن باز واگذار کنید و از او به عنوان نگهدارنده این راهانداز استفاده کنید. راه دیگر این است که نگهداری نرمافزار راه‌انداز را داخل شرکت انجام دهید. هر یک از این دو راه را می‌توانید انتخاب کنید، بسته به اینکه چقدر قصد دارید روی طراحی راهانداز سرمایه‌گذاری کنید و به چه میزان می‌خواهید بر روی آن کنترل داشته باشید. حتی اگر تصمیم به نگهداری نرمافزار در داخل شرکت گرفتید، جامعه متن باز این حق را دارد که نسخه‌های بهتری از این راهانداز را طراحی کند. ارتباط مستقیم با جامعه باعث می‌شود که هر دو طرف کار خود را به درستی انجام دهند و از کدنویسی‌های تکراری جلوگیری شود.

با اینکه تکلیف نرمافزار راهانداز روشن شد، محصول شما هنوز قابل عرضه به مشتریان نیست. آنچه که اکنون شما نیاز دارید، برنامه‌ای است که امکان دیدن تلویزیون را به کاربران بدهد. در این نرمافزار باید امکاناتی از قبیل عوض کردن کanal، تنظیم بلندی صدا و تمامی امکانات یک تلویزیون معمولی موجود باشد. نرمافزارهای پیشرفته‌تر ممکن است امکانات جالب‌تری از قبیل ضبط برنامه‌های تلویزیون، نمایش نام ایستگاه‌های تلویزیونی و رد کردن آگهیهای بازرگانی داشته باشند. اگر سخت‌افزار درست عمل کند و نرمافزار راهانداز هم نوشته شده باشد، برنامه‌نویسان خلاق جامعه‌من بن باز بالاصله راحل‌های خود را ارائه خواهد داد. حتی اگر جامعه نرمافزارهای جذاب‌تری برای سخت‌افزارهای شما طراحی کند، هنوز این امکان برای شما وجود دارد که قابلیتهای بهتری را به آنها اضافه کنید و آنها را به صورت نرمافزار اختصاصی و پشتیبانی شده به فروش رسانید. این تصمیم بستگی به توانایی شما در جذب درآمد بیشتر از محصولات اختصاصی و میزان تمایل شما به عرضه پشتیبانی این محصول دارد. شکل ۱۱-۴ نشان می‌دهد که چگونه ممکن است راهانداز GPL باشد، ولی کاراییهای بالرزش‌تر در قالب نرمافزار نمایش تلویزیون، توسط جامعه‌من بن باز طراحی شود و یا به صورت نرمافزاری اختصاصی عرضه شود.

۱۱-۶-۲- مستندات باز برای واسط سخت‌افزاری

برای اینکه جامعه‌ برنامه‌نویسان در طراحی نرمافزار راهانداز موفق شود، باید مستندات کافی در زمینه عملکرد سخت‌افزار را در اختیار آنها قرار داد. گرچه نمونه‌های زیادی از مهندسی معکوس در جامعه‌من بن باز انجام شده است، اما اگر قرار است آنها راهانداز شما را بنویسند، منطقی نیست که اطلاعات مربوط به سخت‌افزار را از آنها مخفی کنید. منتشر کردن اسناد قابلیتهای داخلی یک محصول برای بسیاری از مدیران قابل قبول نیست، اما اغلب لازم نیست تمامی مستندات سخت‌افزار را منتشر کنید. آنچه که باید منتشر شود فقط اسناد مربوط به کنترل سخت‌افزار است. بهترین راه ایجاد این مستندات این است که گروهی از مهندسان آن را به کمک مستندات اصلی سخت‌افزار تهیه کنند. این مستندات فقط نحوه ارتباط با سخت‌افزار را شرح می‌دهد، بدون اینکه اطلاعات خصوصی آن را که به درد رقبا می‌خورد، آشکار کند.

در مواردی که محصول شما تحت مجوز شرکت دیگری تولید می‌شود، قبل از انتشار این مشخصات باید با تولیدکننده اصلی سخت‌افزار (OEM) مذاکره کنید. در چنین مواردی ممکن است خود تولیدکننده اصلی راهانداز را برای شما طراحی کند.

اگر به هر دلیلی نمی‌خواهید هیچ یک از مستندات را منتشر کنید، می‌توانید یک نرمافزار راه‌انداز مرجع را طراحی کنید و آن را در اختیار جامعه‌من بن باز قرار دهید. این جامعه می‌تواند از این مدل مرجع به عنوان نقطه شروع و جایگزینی برای مستندات سخت‌افزار استفاده کند.

۱۱-۳- همراه کردن نرمافزارهای مورد نیاز با سخت‌افزار

قدم آخر مدل تجاری سخت‌افزار این است که نرمافزارهای متن‌بن را به همراه سخت‌افزار خود عرضه کنید. به یاد داشته باشید هدف شما این است که ارزش بیشتری را به مشتریان ارائه نمائید. در استفاده از یک نرمافزار متن‌بن، لزوم پشتیبانی از آن را نیز در نظر داشته باشد.

۷-۱۱- مدل پایان چرخه زندگی

این مدل نیازمند تحلیل پیچیده‌ایست که بتواند اهداف کاهاش هزینه، و سرعت بخشیدن به پایان زندگی و ایجاد یک جامعه پویا را متعادل و برآورده کند.

وقتی که شرکتی عمر یک محصول را به پایان می‌رساند، باید تأثیرات آن را بر مشتریان در نظر داشته باشیم. مشتریانی که روی محصولی سرمایه‌گذاری کرده‌اند، نسبت به بی‌ارزش شدن سرمایه خود بی‌تفاوت نخواهند بود. آنچه که شرکتها در این شرایط باید در نظر داشته باشند، تأثیر منفی‌ای است که این کار بر وجهه آنها نزد مشتریان می‌گذارد. گرچه اغلب مشتریان منطق پایان دادن به زندگی یک محصول را درک می‌کنند، اما آنها همچنان به خاطر هزینه‌هایی که صرف تهیه آن کرده‌اند، عصبانی خواهند شد.

اگر تولید یک محصول را متوقف می‌کنید، باید پشتیبانی آن را تا چندین سال ادامه دهید تا مشتریان ناراضی نشوند و زمان کافی برای مهاجرت به محصولات جدید را داشته باشند. در مدت پشتیبانی باید تعدادی از مهندسان را مأمور پاسخگویی به این مشتریان و رفع مشکلات آنها کنید. گرچه مشتریان حاضرند هزینه این پشتیبانی‌ها را پرداخت کنند، اما همچنان بخشنی از منابع و انرژی شما بر چیزی متمرکز شده است که جزء اهداف شما نیست.

در چنین شرایطی، متن‌باز بهترین وسیله برای حفظ وجهه نزد مشتریان و پایین بردن هزینه‌های پشتیبانی می‌باشد. بدین ترتیب، می‌توانید تمام انرژی خود را بر اهداف درآمدزای شرکت متمرکز کنید. اگر در مدل تجاري شما اصولاً پشتیبانی در نظر گرفته نشده است، با از طریق متن‌باز کردن، می‌توانید این وظیفه را بر عهده شرکتهای دیگری که متخصص در زمینه پشتیبانی هستند، واگذار کنید. همچنین می‌توان محیطی ایجاد کرد که خود مشتریان پشتیبانی و بهبود آن محصول را ادامه دهند.

اگر قصد دارید کسب و کار کنونی خود را کلاً ترک کنید و در زمینه دیگری آغاز به کار کنید، در این صورت محصول جدیدی نیز وجود نخواهد داشت که مشتریان به فکر مهاجرت به آن باشند. اما در هر کسب و کاری باید وجهه خود را نزد مشتریان و صنعت حفظ کنید و آن را از اولویت‌های خود قرار دهید. در چنین شرایطی بهترین کار این است که نرمافزار خود را تحت مجوزهای متن‌باز در اختیار عموم قرار دهید.

۸-۱۱- ایجاد یک اکوسیستم

در فصل قبل شبیه‌ساز Palm OS را به شما معرفی کردیم. این نرمافزار یکی از اولین نمونه‌های موفق مدل‌های تجاري متن‌باز است. در این مثال نشان داده شد که چگونه یک شرکتی ارزش واقعی خود را تشخیص داد و در ارتباط نزدیکی با جامعه متن‌باز کار کرد. در اینجا بررسی تاریخچه شبیه‌ساز Palm OS خالی از لطف نیست.

وقتی که US Robotics Pilot را به بازار عرضه کرد، بستر برنامه‌نویسی شرکتی آن نرمافزاری بود از شرکت Metrowerks. شرکت Metrowerks در طراحی ابزار برنامه‌نویسی برای محیط‌های توکار تخصص داشت. برای برنامه‌نویسان تجاري این ابزار قابل قبول بود، اما اکثر

برنامهنویسان ابزار برنامهنویسی می‌خواستند که تحت پلتفرم PC نیز قابل اجرا باشد. برنامهنویسان وقتی که برای دستگاه‌های Embedded PDAها برنامهنویسی می‌کنند، ترجیح می‌دهند برنامه خود را در همان سامانه‌ای که برنامهنویسی روی آن صورت می‌گیرد، آزمایش کنند. بنابراین باید امکان شبیه‌سازی PDA در محیط دیگری مانند PC باشد. این شبیه‌سازی علاوه بر اینکه آزمایش نرم‌افزار را راحت‌تر می‌کند، خلاقیت و سرعت برنامهنویس را نیز بالا می‌برد. بدون شبیه‌ساز، برنامهنویس باید پس از هر تغییری برنامه را به یک PDA واقعی برای آزمایش منتقل کند. قابل ذکر است که در PDAها به دلیل محدود بودن امکانات، برنامهنویسی بسیار مشکل است. بنابراین باید این کار در سامانه‌های دیگری مانند PCها صورت گیرد. گرچه ابزار برنامهنویسی Metrowerks بسیار کامل و جامع بود، اما آنقدر گران بود که افراد عادی قادر به خرید آن نبودند. از طرف دیگر ابزار این شرکت فقط تحت سامانه‌های مکیتاش قابل استفاده بود. از آنجایی که مکیتاش و پایلوت از نسل مشترکی از پردازنده‌ها (موتورولا ۶۸۰۰) استفاده می‌کردند، شبیه‌سازی پایلوت در مکیتاش بسیار راحت بود. به تدریج جامعه‌ای از برنامهنویسان پایلوت (امروزه Palm نامیده می‌شود) تشکیل شد. هدف آنها این بود که ابزار برنامهنویسی برای پایلوت بسازند که به راحتی در پلتفرم PC قابل استفاده باشد. در آن زمان مفاهیم متن‌باز به این گستردگی مطرح نشده بود و بسیاری از آن اطلاع نداشتند. این برنامهنویسان مجبور بودند تمامی اجزای شبیه‌ساز مورد نظر را به کمک مهندسی معکوس بسازند. این اجزا یکی پس از دیگری کنار یکدیگر قرار گرفتند. یکی از برنامهنویسان توانست فرمت برنامه‌های اجرایی Palm را بدست آورد. برنامهنویس دیگری ابزاری برای کامپایل کردن متابع رابط کاربری نوشت. برنامهنویس دیگری هم کمپایلرهای متن‌باز را برای ساختن (کامپایل) و غلطیابی برنامه‌های Palm اصلاح کرد. در همین حال، گرگ هیوجیل سرگرم بررسی شبیه‌سازهای Pilot تحت پلاتفرمهای دیگر بود و سرانجام CoPilot نوشته شد. گرچه این وقایع بسیار ساده به نظر می‌رسند، اما همین وقایع ساده منجر به تولید کاملترین شبیه‌ساز Pilot تحت PC شدند. این شبیه‌ساز گرچه به زیبایی نسخه Metrowerks نبود، اما از نظر امکانات چیزی از آن کمتر نداشت.

مشکل دیگری که گرگ باید حل می‌کرد مسئله ROM دستگاه بود. در PDAها، سامانه‌عامل در حافظه فقط خواندنی یا ROM قرار دارد. آنچه که گرگ طراحی کرده بود، فقط یک سخت‌افزار PDA را شبیه‌سازی می‌کرد، اما برای اینکه بتوان از آن شبیه‌سازی استفاده کرد، باید سامانه‌عامل PDA نیز به آن اضافه می‌شد. او توانست به کمک سایر برنامهنویسان، ابزاری برای خواندن ROM دستگاه بنویسد. به کمک این ابزار می‌توانستند ROM دستگاه را به صورت قطعه‌هایی دریافت و به شبیه‌ساز اضافه کنند (تجهیز کنید که امروزه قوانینی وجود دارد که اجازه این کار را به شما نمی‌دهد). بدین ترتیب پلاتفرم شبیه‌سازی شده کامل شد. اما از آنجایی که US Robotics مجوز انتشار کد ROM را نداده بود، هر کاربری باید ROM دستگاه خود را

استخراج و در شبیه‌ساز استفاده می‌کرد. این بدین معنا بود که اگر قصد نوشتمن برنامه‌ای برای یک PDA داشتید، باید یک دستگاه از آن را تهیه می‌کردید.

از اینجا نقطه پویایی Pilot آغاز شد. تعداد برنامه‌های Pilot به طور غیرقابل تصویری افزایش یافت. تعداد برنامه‌نویسان این دستگاه نیز افزایش یافت و یک اکوسیستم متکی به خود ایجاد شد. در آن زمان 3Com (و سپس USRobotics) از درک شرایط پیش آمده عاجز بودند. از یک طرف محرومگی اطلاعات اختصاصی دستگاه از بین رفته بود و از طرف دیگر این جامعه برنامه‌نویسان با نوشتمن برنامه‌های متنوع، ارزش این دستگاه را بالاتر برده بودند. وقتی که Palm از این شرکتها مستقل شد، در آستانه یک تصمیم مهم قرار گرفت: آیا از این جامعه استقبال کند یا با آن بجنگ و کنترل PDA را دوباره بدست آورد؟ Palm تصمیم به استقبال از این جامعه گرفت. اکنون Metrowerks ابزار برنامه‌نویسی خود را هم برای مکیتاش و هم برای PC عرضه کرده است. تولیدکنندگان نرمافزار تجاری همچنان از این ابزار برای نوشتمن برنامه‌های این دستگاه استفاده می‌کند. اما افراد عادی ابزار دیگری در اختیار دارند که رایگان است. Palm و Metrowerks ارزش CoPilot را درک کردن و مجوز انتشار آن را از گرگ و سایر برنامه‌نویسان گرفتند. سپس نام آن را به شبیه‌ساز (POSE) PalmOS تغییر دادند و تحت مجوز GPL منتشر کردند. امروزه POSE را همراه نسخه تحت PC ابزار برنامه‌نویسی خود عرضه می‌کند. این برنامه از سایت Palm نیز به طور رایگان قابل دریافت است.

شرکت Palm به این نتیجه درست رسید که ابزار با ارزشی است که مجموعه ابزار برنامه‌نویسی Palm را تکمیل می‌کند. دقت کنید که صرف عرضه این ابزار به مشتریان ارزشی ندارد. بلکه ارزش در رایانه‌ای قابل حمل نهفته است که طیف گسترده‌ای از نرمافزارهای کاربردی برای آن وجود دارد. یعنی ارزش این دستگاه با افزایش تعداد برنامه‌ها بالا می‌رود و POSE این مهم را تحقق بخشیده است.

امروزه شرکت Palm و جامعه برنامه‌نویسان به طور مشترک روی POSE کار می‌کنند. Palm وظیفه نگهداری این برنامه را بر عهده عده‌ای از برنامه‌نویس خود قرار داده است. اما جامعه برنامه نویسان، آن را بهبود می‌دهد و اشکالات آن را برطرف می‌کند. بدین ترتیب Palm مسیر توسعه POSE را در جهت بالا بردن ارزش سخت‌افزارهایش هدایت می‌کند. بدین ترتیب این شرکت توانسته است هزینه‌های خود را در سطح پایینی نگه دارد و اکوسیستم بالارزشی ایجاد نماید. امروزه Palm برنامه موجود در ROM هر دستگاه را آزادانه در اختیار عموم قرار می‌دهد و نسخه‌های جدیدتر سامانه‌عامل را در سایت خود می‌گذارد.

آنچه که دیدیم، مثال بسیار خوبی بود از یک مدل تجارتی موفق. در این مدل تجارتی نشان داده شد برای بهره برداری از متن‌باز، باید بدانیم ارزش واقعی محصول ما در کجای آن قرار دارد. نکات قابل توجه در این مدل تجارتی عبارتند از:

- شرکت Palm به درستی ارزش آنچه را که به مشتریان عرضه می‌کرد، فهمید: یک دستگاه قابل حمل با تعداد بسیار زیادی برنامه.

- جامعه برنامهنویسان و Palm اکوسیستم با ارزشی درست کردن که باعث ترویج Palm و بالا رفتن ارزش آن شد.
 - جامعه برنامهنویسان یکی از ابزارهای کلیدی زنجیره ارزش Palm را طراحی کرد که به برنامهنویسان این امکان را می داد که برنامه های Palm را در یک PC بنویسند و آزمایش کنند.
 - Palm همچنان در توسعه POSE با این جامعه همکاری می کند. این همکاری باعث می شود شبیه ساز همان مسیری را طی کند که Palm می خواهد و در عین حال داشن برنامه نویسان در زمینه این دستگاه نیز بالا می رود.
 - عرضه کردن برنامه موجود در ROM نوعی از عرضه مستنداتی است که جامعه برنامه نویسان برای موفقیت در کار خود، به آن نیاز دارند. البته این شرکت مستندات دیگری نیز در اختیار برنامه نویسان قرار داده است.
- هیچ کس در US Robotics ، 3COM و Palm تصویر چنین وقایعی را نمی کرد. اکنون شما این شانس را دارید که از آنها که این مسیر را طی کرده اند یاد بگیرید و آموخته های خود را در سازمان خود اعمال کنید.

۹-۱۱- جمع بندی

شکست یا موفقیت یک مدل تجاری بستگی به این دارد که چگونه اجرا شود. فهمیدن یک مدل تجاری جای تحلیل جامع تجاری را نمی گیرد. شما هنوز هم باید بازار و نیازهای مشتریان خود را بشناسید و همه عناصر را برای ایجاد یک کسب و کار سودآور را نیاز دارید. متن باز فقط یکی از این عناصر است که دروازه هایی را برای ساختن یک مدل تجاری جدید یا بهبود مدل کنونی به روی شما می گشاید.

اکنون که شما طرحهای تجاری قابل استفاده را به خوبی درک کرده اید، باید نحوه پیاده سازی متن باز را در سازمان خود بیاموزید.

در فصل بعد در مورد تصمیم گیری در زمینه استفاده و عرضه محصولات متن باز بحث خواهد شد.

فصل دوازدهم

پیاده‌سازی متن‌باز در کسب و کار

در سه فصل گذشته جزئیات نحوه کار متن‌باز شرح داده شد و همچنین تأثیرات اقتصادی آن بر صنعت نرم‌افزار بررسی شد. در همین راستا چندین مدل تجاري برای شرکتهای تولیدکننده نرم‌افزار معرفی شد. با فرض اینکه شما قصد پیاده‌سازی یکی از این مدل‌های تجاري نوین را دارید، باید تغییرات پایه‌ای مورد نیاز در کسب و کار خود را معین کنید. در این فصل و فصل بعد فرآیندهای مورد نیاز برای تصمیم‌گیری درست در این زمینه شرح داده خواهد شد. اهداف این فصل عبارتند از:

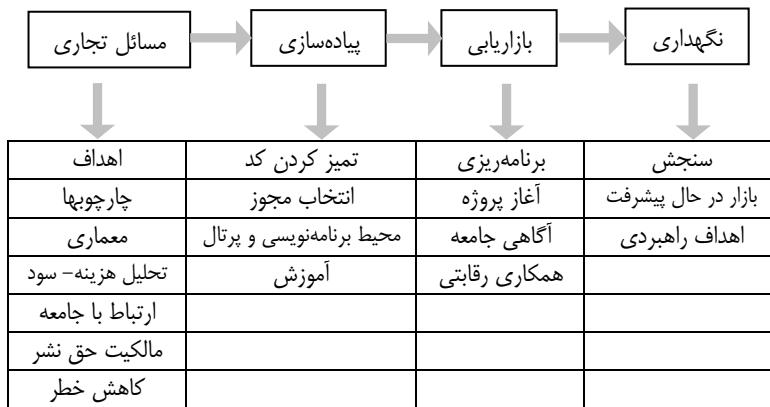
- بررسی دقیق دلایل متن‌باز کردن یک نرم‌افزار
- تهیه لیستی از کارهایی که موقع متن‌باز کردن یک نرم‌افزار اختصاصی باید انجام داد
- بررسی مشکلات به کارگیری متن‌باز در کسب و کار
- بررسی مسائل قانونی استفاده از متن‌باز

این فصل هم برای مدیران فناوری اطلاعات و هم برای شرکتهای تولیدکننده نرم‌افزار مفید است. مدیران فناوری اطلاعات دوست دارند از نرم‌افزارهای متن‌باز به عنوان راهی برای کاهش هزینه‌ها استفاده کنند. اما برای شرکتهای تولیدکننده نرم‌افزار هر دو مقوله آوردن نرم‌افزار به داخل شرکت و عرضه نرم‌افزار متن‌باز قبل توجه خواهد بود.

۱۲-۱- عرضه نرم‌افزار متن‌باز

منتظر از عرضه نرم‌افزار متن‌باز در اینجا این است که نرم‌افزاری که در حال حاضر به صورت اختصاصی به فروش می‌رود، تحت یک مجوز متن‌باز عرضه شود. برای انجام موفق آمیز این کار، باید مراحل نمایش داده شده در شکل ۱-۱۲ را دنبال کنید.

همان گونه که در شکل ۱-۱۲ می‌بینید، عرضه نرم‌افزار به جامعه متن‌باز فرآیند آسانی نیست. برای انجام این کار باید برنامه‌ریزی و تحلیل دقیقی صورت گیرد. البته پژوهه‌ها متفاوت هستند و میزان پیچیدگی آنها با یکدیگر فرق می‌کند. در برخی از موارد کل پرونده ممکن است در مدت کوتاهی انجام شود. اما به هر حال باید حداقل تأمیلی در عناصر دخیل در این فرآیند صورت گیرد. یک خلبان هر چقدر هم باتجربه باشد، یکسری موارد را قبل از پرواز چک می‌کند.



شکل ۱-۱۲: فرآیندهای متن‌باز کردن

۱-۱-۱۲- مسائل تجاری

اولین قدمی که باید برداشته شود این است که مسائل تجاری این فرآیند دقیقاً بررسی شود و این مسئله مشخص شود که آیا انجام این کار منطقی است یا خیر. در شکل ۱-۱۲ عناصر کلیدی مربوط به مسائل تجاری مشخص شده‌اند. از آنجایی که شما در دنبال کردن این فرآیند مبتدی هستید بررسی هر یک از این عناصر خالی از لطف نیست.

اهداف

قبل از آغاز هرگونه پیاده‌سازی باید دلایل منطقی برای آغاز پروژه‌ها به صورت متن‌باز داشته باشیم و همه افراد درگیر در آن پروژه از آن اهداف به خوبی آگاه باشند. در اینجا چند مثال از اهداف مختلف یک کسب و کار آمده است. اکثر این مثالها با مدل تجاری مطرح شده در فصل قبل همخوانی دارد.

- بی‌ارزش کردن یک محصول رقابتی- اگر رقیب شما محصولی داشته باشد که از محصول شما بسیار بهتر است یا شما اصلاً آن را عرضه نمی‌کنید، انگیزه کافی برای استفاده از این روش را دارید. بنابراین هدف شما این است که سرعت کاهش ارزش محصول رقیب را افزایش دهید و جریان درآمد و سود او را قطع کنید. اما این نکته را در نظر داشته باشید که با انجام این کار بر تمام بازار که خود شما هم عضوی از آن هستید، تأثیر می‌گذارد.
- قدرت نفوذ- اگر فناوری در انحصار شما راهی برای ارتقاء محصولات دیگران است، می‌توانید از متن‌باز کردن به عنوان روشی برای مطرح شدن خود و انتخاب آن فناوری به عنوان استاندارد استفاده کنید.
- کاهش هزینه‌ها- در شرایط خاص، کار مشترک با جامعه متن‌باز هزینه‌های تولید نرم‌افزار را به طور قابل توجهی کاهش می‌دهد. در چنین مواردی اگر در حال عرضه

محصولی بزرگ هستید، متن باز کردن برخی از اجزاء آن سرعت توسعه آن را بیشتر و هزینه آن را کمتر می‌کند.

- سختافزار- همان گونه که در مدل تجاری سختافزار اشاره کردم، اگر سختافزار خاصی را عرضه می‌کنید، در بهترین شرایط برای کار با جامعه متن باز هستید.
- خدمات و پشتیبانی- اگر مدل تجاری شما بر اساس ارائه خدمات بنا شده است، توصیه محصولات نرم‌افزاری دیگران به مشتریان، منطقی به نظر می‌رسد.
- طرح خروج- اگر در حال ترک کسب و کار کنونی خود هستید، متن باز کردن محصولات و مستندات می‌تواند بسیار مفید باشد.
- شرکت- متن باز به شرکت و همکاری تشویق می‌کند. شما می‌توانید از متن باز به عنوان راهی برای کار کردن با شرکاء و رقبا بر روی پروژه‌های سنگین، استفاده کنید.
- انتشار فناوری از رده خارج- اگر شرکت شما فناوری را توسعه داده است که امروزه جزو اهداف استراتژیک آن نیست و قصد ندارید هزینه ثبت آن را پرداخت کنید، می‌توانید آن را تحت مجوزهای متن باز منتشر کنید.

به همین ترتیب نیز دلایل زیادی هم وجود دارد برای اینکه متن باز کردن به سوددهی نمی‌رسد. تعدادی از این دلایل عبارتند از:

- محصول یکه تاز- اگر محصول شما حرف اول را در میان محصولات موجود می‌زند و مانع ورود محکمی برای رقباست، نباید آن را متن باز کنید.
- محصول منسخ شده- پایان عمر یک محصول لزوماً به معنای منسخ شدن آن نیست. اگر احساس می‌کنید که محصول شما از دید شما و مشتریانتان دیگر قابل استفاده نیست، بلاfacile عرضه آن را متوقف کنید. در چنین شرایطی، زنده نگهداشتن محصول از طریق متن باز کردن، به هیچ وجه توصیه نمی‌شود.
- سود منفی- اگر برسیهای شما نشان دهد که سود متن باز از ضررهای آن کمتر است، نباید به سراغ آن بروید.
- عدم تمرکز منابع- اغلب پس از عرضه محصول به جامعه متن باز نیز مجبور به صرف قسمتی از منابع داخلی خود برای آن هستید. اگر این کار باعث عدم تمرکز منابع سازمان شما و هدف اصلی شود، باید از آن اجتناب کنید.
- خطرات مالکیت معنوی- وکیل شرکت باید مدتی با شما کار کند تا بفهمد چه کسی مالکیت معنوی هر محصول را در اختیار دارد. اگر خطرات قانونی مالکیت معنوی زیاد باشد، در این صورت نباید این گونه پروژه‌ها را ادامه دهید.
- رقابت با جامعه- رقابت با پروژه متن باز موجود به هیچ وجه توصیه نمی‌شود و فقط وجهه شما را در جامعه متن باز خراب می‌کند. اگر پروژه‌ای مشابه پروژه شما در این جامعه در حال توسعه است، باید به جای رقابت با آن، تلاش کنید که به آن بپیوندید و در بهبود آن تلاش کنید. البته در مواردی که پروژه‌ای اهدافی غیر از اهداف شما را

دنبال می‌کند، رقابت با آن مشکلی ندارد. برای اینکه درست تصمیم بگیرید، باید از راهنماییهای جامعهٔ متن‌باز استفاده کنید.

- فناوری جالب- بسیاری از مهندسان فکر می‌کنند چون فناوری جالبی طراحی کرده‌اند، بلافصله باید آن را متن‌باز کنند. دیدگاه آنها این است که تشویقهای مردم باعث پیشرفت بهتر پروژه می‌شود. اما در اغلب موارد هزینه سنگین پروژه بر تشویقهای زودگذر غالب می‌شود و اگر پروژه به نتیجه مطلوب نرسد، واکنشهای منفی، به موارد فوق افروزه می‌شود.

وقتی که اهداف کلیدی شرکت شما برای یک پروژه مشخص شد، می‌توانید قدم بعدی را در زمینهٔ مسائل تجاری بردارید.

معیارها

پس از تعیین اهداف پروژه، باید معیارها را مشخص کنیم. این معیارها برای سنجش میزان موفقیت پروژه لازمند. تعدادی از معیارها عبارتند از:

- رقیبی که باعث کاهش قیمت محصول شده است
- تعداد برنامه‌نویسانی که به طور فعال در پروژه نقش دارند
- تعداد کارهای ارائه شده در پروژه
- تعداد کاربرانی که از پروژه استفاده می‌کنند
- عناصری از اکوسیستم که حول پروژه شما در حال گرفتن هستند
- کاهش هزینه‌های توسعه به کمک عرضه به موقع محصول

این معیارها بستگی به مشخصات پروژه دارند. شما باید بتوانید نقطهٔ موفقیت را تعریف و زمان رسیدن به آن را تعیین کنید.

معماری

در فاز معماری از منابع فنی‌ای که در پروندهٔ تصمیم‌گیری دخیل بودند، استفاده می‌شود. متخصصان فناوری باید معماری پروژه متن‌باز پیشنهاد شده را معین کنند. آنچه که تیم فنی باید مشخص کند، عبارتند از:

- آیا پیمانه‌های مختلف نرم‌افزار به خوبی از یکدیگر مجزا شده‌اند؟
- آیا می‌توان نرم‌افزار را به اجزای متن‌باز و اختصاصی تقسیم کرد؟
- آیا این پروژه متن‌باز می‌خواهد پلی برای پروژه‌های اختصاصی باشد؟
- آیا رابطه‌ای مناسب برای اتصال این دو پروژه در نظر گرفته شده است؟
- آیا می‌توان این نرم‌افزار را از سامانه‌های مدیریتی کنونی شرکت مجزا کرد؟
- آیا مالکیت معنوی تمامی نرم‌افزار حاصل در اختیار شماست؟ در غیر این صورت آیا شما مجوز انتشار آنها را تحت مجوزهای متن‌باز دارید؟ آیا می‌توانید عناصر مشکل‌دار نرم‌افزار را عوض کنید؟

- آیا کدی که قرار است عرضه شود، قبلاً به نام کسی ثبت شده است؟ حتی اگر این کد به نام شما و شخص دیگری ثبت شده باشد، ممکن است شما اجازه انتشار آن را تحت مجوزهای متن باز نداشته باشید.

پروژه‌ها گوناگون هستند و هر نرم‌افزار معماری منحصر به فردی دارد. اگر اهداف پروژه را مشخص کرده باشید، تیم فنی می‌تواند بر این اساس، محصول قابل قبولی برای شما آماده کند.

تحلیل هزینه و سود

بسیاری از مردم فکر می‌کنند فرآیند انتقال یک نرم‌افزار به مجوزهای متن باز، فقط مستلزم بسته‌بندی آن نرم‌افزار و قرار دادن آن در اینترنت برای عموم است.

اما واقعیت این است که فرآیند متن باز کردن می‌تواند بسیار پرهزینه باشد. در هر تحلیل مالی باید اهداف و پویایی پروژه در نظر گرفته شود. برخی از مواردی که باید در تحلیل هزینه پروژه در نظر گرفته شود، عبارتند از:

- ایجاد یک مخزن نرم‌افزار جدید
- تصفیه کدهای موجود و حذف بخش‌های اضافه و توضیحات برنامه (COMMENT)
- برنامه‌های تبلیغاتی برای اعلام متن باز شدن پروژه و ایجاد جامعه‌ای از برنامه‌نویسان حول آن
- نگهداری نرم‌افزار (با فرض اینکه شما می‌خواهید همچنان نگهدارنده آن نرم‌افزار باشید).
- انتشار مستندات: شامل معماری، طراحی و راهنمای کاربر
- ایجاد پرتال برای آن نرم‌افزار برای همکاری بهتر جامعه با پروژه
- آموزش نرم‌افزار: هزینه آموزش نرم‌افزار بستگی به میزان پیچیدگی آن دارد
- تحمل هزینه دریافت محصول از اینترنت: برای اینکه نرم‌افزار را در اختیار عموم قرار دهید، باید از کارسازهایی برای دریافت آن استفاده کنید و هر چه میزان دریافت یک پروژه از اینترنت بیشتر باشد، شما باید برای مهار ترافیک ایجاد شده سرمایه‌گذاری کنید.

طبعیاً انتظار دارید که این هزینه‌ها را درآمدی جبران کند. این درآمد ممکن است به صورتهای زیر باشد:

- کاهش هزینه‌های برنامه‌نویسی داخل شرکت
- کاهش هزینه‌های پشتیبانی
- شناخته شدن در بازار
- افزایش درآمد محصولات همراه (سخت‌افزاری یا نرم‌افزاری)
- ضرر کردن رقبا

این تحلیل هزینه و سود، اطلاعات کافی برای تصمیم‌گیری درست را در اختیار شما قرار می‌دهد.

ارتباط با جامعه

ارتباط با جامعه این یکی از مهمترین بخش‌های فرآیند تصمیم‌گیری است که متأسفانه اغلب نادیده گرفته می‌شود. با اینکه بسیاری از حامیان متن باز معتقدند که همه نرم‌افزارها باید در اختیار عموم قرار گیرند، همان‌گونه که دیدید این کار واقعاً هزینه‌بر است. فقط در صورتی متن باز کردن پروژه منطقی است که بتوان هزینه‌های آن را از جای دیگری جبران کرد. ممکن است نرم‌افزار شما به گونه‌ای باشد که متن باز کردن آن توجه جامعه را به خود جلب نکند. در این صورت این اقدام شما برای هیچ کس اهمیت نخواهد داشت و هیچ کس در توسعه آن شرکت نخواهد کرد. شما باید احتمال ایجاد جامعه برنامه‌نویسان برای پروژه آتی خود را پیش‌بینی کنید. اگر در حال افزودن عملکردی به یک پروژه موجود هستید، می‌توانید با سرمداران جامعه برنامه‌نویسان درباره این پروژه صحبت کنید و نظر آنها را در زمینه میزان تمایل افراد جامعه به شرکت در این پروژه جدید جویا شوید.

اگر قصد دارید این جامعه را به کمک مشتریان خود تشکیل دهید، باید از آنها نظرسنجی کنید تا از میزان تمایل آنها به شرکت در این جامعه برنامه‌نویسان آگاه شوید.

در نهایت اگر قصد دارید جامعه جدیدی برای یک فناوری جدید ایجاد کنید باید در زمینه شناساندن آن پروژه به برنامه‌نویسان تلاش کنید و در عین حال نباید هزینه‌های اجرای چنین پروژه‌ای را نیز دست کم بگیرید. برای جذب برنامه‌نویسان خبره باید در فکر ایجاد انگیزه در آنها نیز باشیم.

مالکیت حق کپی

گرچه من قبلاً هم به مسأله مالکیت حق کپی اشاره کرده‌ام، لازم به یادآوریست که این مسأله مهمترین بخش از فرآیند تصمیم‌گیری است. نه تنها مالکیت کدهای موجود را باید در نظر بگیرید، بلکه گرفتن مجوز کپی از کارهای ارائه شده توسط جامعه نیز حائز اهمیت است. فرآیند تصمیم‌گیری تا حدی هم بستگی به مدل تجاری دارد که انتخاب کرده‌اید. اگر از فصل قبل به یاد داشته باشید، شما می‌توانید نرم‌افزار خود را تحت دو مجوز عرضه کنید: یکی متن باز و دیگری اختصاصی. اگر مدل تجاری مجوز دوگانه را انتخاب کرده‌اید، باید مراحلی را برای دریافت کارهای انجام شده از کسانی که قصد دارند حق کپی کارشان را به شما واگذار کنند، طراحی کنید یا اینکه مجوزی برای آن در نظر گیرید که امکان انشعاب به مجوزهای فرعی برنامه‌نویسان را داشته باشید.

کاهش خطر

مسلمان شما دوست دارید پروژه را به بهترین نحو به انجام برسانید، اما جنبه‌های منفی اجرای پروژه را نیز باید در نظر داشته باشید. باید احتمال درست پیش نرفتن کارها را بدھید و برای چنین شرایطی آمادگی کافی داشته باشید. در طرح کاهش خطرات باید موارد زیر را در نظر داشته باشیم:

- اگر جامعه برنامه‌نویسی حول پروژه شما ایجاد نشد، چه می‌کنید؟ چقدر منتظر ایجاد این جامعه می‌شوید؟ اگر جامعه علاقه‌ای برای شرکت در پروژه نشان نداد، آیا باز هم برای تبلیغ پروژه سرمایه‌گذاری می‌کنید؟ بر اساس چه معیاری ممکن است پروژه را متوقف کنید؟
 - اگر هزینه‌های پروژه از ارقام پیش‌بینی شده تجاوز کرد، چه می‌کنید؟ آیا برنامه‌ای برای جبران این هزینه‌ها دارد؟
 - اگر کسی در زمینهٔ مالکیت معنوی پروژه ادعا کرد، چه می‌کنید؟ آیا اسناد کافی در این زمینه در دست دارد؟
 - اگر پروژه دیگری در رقابت با پروژه شما آغاز به کار کرد چه می‌کنید؟ آیا تمهیداتی برای جلوگیری از اجرای پروژه مشابه اندیشیده‌اید؟
 - اگر جامعه متن‌باز شروع به اجرای پروژه مشابهی کرد که اهداف شما را تحت الشعاع قرار می‌دهند، چه می‌کنید؟ واکنش شما چه خواهد بود؟ آیا اصلاً اهمیتی برای شما دارد؟
 - اگر میزان دریافت نرم‌افزار از سایت شما چنان ترافیک را بالا ببرد که عملاً دریافت آن امکان‌پذیر نباشد، چه می‌کنید؟ چه تمهیداتی برای بالابردن پهنه‌ای باند سایت در چنین شرایطی اندیشیده‌اید؟
- در این زمینه وقت زیادی بگذرید و هر آنچه که ممکن است مشکل‌ساز شود، پیش‌بینی کنید. برای این کار، لزومی ندارد بدینانه به پروژه نگاه کنید، بلکه عاقلانه و دقیق به آن بنگرید.

۱۲-۱-۲- پیاده‌سازی

پس از اینکه فرآیند تحلیل مسائل تجاری را پشت سر گذاشتید، اطلاعات کافی را برای تصمیم‌گیری در زمینهٔ انجام یا عدم انجام پروژه خواهید داشت. اگر تصمیم به آغاز پروژه گرفتید، به فاز پیاده‌سازی می‌رسید. در این فاز باید طرحها را عملی کرد.
تمیز کردن کد برنامه

پیاده‌سازی با تمیز کردن کد آغاز می‌شود تا کدی حاصل شود که قابل عرضه به عموم باشد. موارد زیر باید در تمیز کردن کد در نظر گرفته شوند:

- آنچه که عرضه می‌کنید به سادگی قابل کامپایل باشد و تمامی کارآیی‌ها در آن گنجانده شده باشد - جامعه متن‌باز نسبت به نرم‌افزارهایی که ناکارا باشند، واکنش منفی نشان می‌دهد. این بدان معنا نیست که نرم‌افزار باید خالی از اشکال باشد، بلکه باید چنان باشد که کاربران نحوه عملکرد آن را متوجه شوند.
- کد دارای ساختار و توضیحات - اگر این کد را از یک پروژه بزرگتر استخراج می‌کنید، ایجاد ساختار در آن و اضافه کردن توضیحات به آن کار بسیار پیچیده‌ای خواهد بود. اما فراموش نکنید که برنامه‌نویسان خارج سازمان باید قادر به درک نحوه عملکرد کد

برنامه باشند. هر چه درک کد شما آسانتر باشد، علاقمندی به کار بر روی آن بیشتر می‌شود.

- تصفیه کد- باید مطمئن شوید که کد شما عاری از هرگونه اطلاعات خصوصی است. این اطلاعات معمولاً به صورت نام کارمندان، اسم رمز محصولات و محصولات آینده است. در میان توضیحات برنامه، ممکن است عبارتهای موهن به کار رفته باشد که برای حفظ آبروی خود باید آنها را با دقت بیاید و حذف کنید.
- وضعیت مالکیت معنوی کد را روشن کنید- شما باید از اینکه کد پروژه تماماً داخل سازمان نوشته شده است و مجوز انتشار آن را دارید، مطمئن شوید. در صورتی که کدی توسط پیمانکاران شما نوشته شده باشد، باید مجوز انتشار آن را از آنها دریافت کرده باشید تا بتوانید آن کد را به سایر کدهایی که متن باز هستند، اضافه کنید. در این مرحله باید با یک حقوقدان با تجربه در زمینه مالکیت معنوی مشورت کنید. از موارد دیگری که باید به همراه کد ارائه کنید، راهنمای تفضیلی کامپایل کردن کد و تشریح هرگونه وابستگی برنامه به کتابخانه‌ها می‌باشد. اگر می‌خواهید جامعه‌ای فعال در زمینه پروژه شما ایجاد شود، باید کد عرضه شده از کیفیت بالایی برخوردار باشد. شما و مهندسین شما باید این مسئله را مدنظر داشته باشید.

انتخاب مجوز

این مرحله را باید با کمک مشاور حقوقی خود پیش بیرید. در فصل ۳ اشاره شد که بیش از ۳۰ مجوز مورد قبول OSI وجود دارد. باید یکی از این مجوزهای موجود را برای پروژه انتخاب کنید. اولین قدم این است که در زمینهٔ دو طرفه بودن مجوز (بدین معنا که هر کاری که روی پروژه انجام می‌شود، باید به جامعه عرضه شود) تصمیم‌گیری کنید. اگر قصد دارید جامعه‌ای ایجاد کنید که انگیزه برای نوآوری و رشد پروژه داشته باشد، باید از این گونه مجوزها و مجوزهای دوطرفه استفاده کنید که رایج‌ترین آنها GPL و LGPL است. اما اگر می‌خواهید این امکان را در اختیار دیگران بگذارید که از نرمافزار شما برای تولید محصولات اختصاصی استفاده کنند، باید از مجوزهای BSD یا MIT استفاده کنید. البته همان گونه که قبلاً اشاره کردم، گزینهٔ مجوز دوگانه را نیز همچنان در اختیار دارید.

صرفنظر از نوع مجوزی که انتخاب می‌کنید، باید برای تمامی پروژه‌های متن باز خود، یک خط مشی ثابت تعیین کنید. منظور از خط مشی ثابت این نیست که همه پروژه‌ها تحت یک مجوز عرضه شوند، بلکه باید برای پروژه‌های از دسته مجوزهای مشابهی استفاده شود. بدین ترتیب افراد خارج از سازمان شما می‌دانند چه انتظاراتی باید داشته باشند. این خط مشی باید بخشی از سیاست کلی سازمان شود.

محیط برنامه‌نویسی

اکنون باید محیط برنامه‌نویسی، ابزار و فرآیندهای کار با جامعه را معین کنید. این مرحله ممکن است به سادگی قرار دادن پروژه در سایت و سرزنشدن به آن باشد. اما این راه خوبی برای

ایجاد یک جامعه برنامهنویسان پویا نیست. اکثر برنامهنویسان از ابزار رایج و استفاده شده در سایر پروژه‌های متن‌باز استفاده خواهد کرد. تعدادی از این ابزار در فصلهای قبل توضیح داده شده‌اند. اما می‌توان از ابزار ساده دیگری نیز برای به اشتراک گذاشتن اطلاعات استفاده کرد، مانند پست الکترونیک. بستگی به نوع پروژه، می‌توانید از گروههای خبری نیز برای تسهیل ارتباط بین برنامهنویسان استفاده کنید.

همچنین درباره محل نگهداری پروژه نیز باید تصمیم‌گیری کنید. می‌توانید آن را روی کارسازهای خودتان قرار دهید یا از میزبانی (Hosting) شرکت‌های ثالث استفاده کنید. در زمینه میزان ترافیک این کارسازها، باید به برنامه‌های بازاریابی خود مراجعه کنید.

آموزش

اغلب آموزش را جزو مراحل متن‌باز کردن نرم‌افزار در نظر نمی‌گیرند و این اشتباه است. آموزش باید در برنامه‌های بازاریابی شما در نظر گرفته شده باشد. اکثر برنامهنویسان قادر به شرکت در کلاسهای آموزشی یا مسافت به شهر دیگری برای آموزش نیستند. در صورتی که قصد تشکیل یک جامعه‌ای متشکل از مشتریان خود دارید و آموزش محصول را به آنها و عده داده‌اید، باید از طریق آموزش رسمی آنها را تعلیم دهید.

البته امکان آموزش‌های الکترونیکی، از قرار دادن مطالب آموزشی در سایت گرفته تا سمینارهای گسترشده اینترنتی نیز در اختیار شماست.

۱۲-۳-۱- بازاریابی

برای کسانی که تصور می‌کنند متن‌باز کردن نرم‌افزار کار ساده‌ایست، بازاریابی نرم‌افزار نیز مسخره به نظر می‌رسد. اما پروژه‌های موفق آنها بی‌هی هستند که برنامه‌های مستند و حساب شده‌ای برای بازاریابی داشته باشند. در اینجا بخش‌های اصلی برنامه بازاریابی را بررسی می‌کنیم:

برنامه‌ریزی برای آغاز پروژه

از مهمترین بخش‌های بازاریابی، برنامه‌ریزی برای آغاز یک پروژه است. در این مرحله باید با اعضای کلیدی جامعه برنامهنویسان مشورت کنید. اگر بتوانید از این اعضای کلیدی در اعلان عمومی برای آغاز پروژه نام ببرید به آغاز پروژه بسیار کمک خواهد کرد. اگر شرکای مشهوری هم دارید باید از آنها برای آغاز پروژه استفاده کنید. در مورد پروژه‌هایی که مربوط به یک صنعت خاص می‌شوند، باید از فرصت‌های پیش‌آمدۀ در سمینارها و واقعیّات این صنعت برای معرفی پروژه خود استفاده کنید.

آگاهی جامعه برنامه‌نویسان

باید آگاهی جامعه برنامهنویسان را در زمینه پروژه بالا ببرید تا انگیزه شرکت در پروژه در آنها تقویت شود. عبارت «جامعه برنامه‌نویسان» یا به طور اختصار جامعه، یک عبارت کلی است که به هر گروهی از برنامهنویسان اطلاق می‌شود. مسلماً اگر شما قصد دارید برنامه‌ای برای مدیریت منابع انسانی تولید کنید، مسلماً نیازی به هکرها ندارید.

همکاری رقابتی

علت اصلی کار با رقبا این است که یک شرکت به تنها یی قادر نیست هزینه‌های یک پروژه را پرداخت کند. به خصوص وقتی که پروژه به گونه‌ای است که کل جامعه از نتایج آن بهره می‌برند. در برخی صنایع کار با رقبا امری عادی و رایج است، ولی در برخی دیگر از صنایع پیوستن به رقبا در اجرای یک پروژه عجیب به نظر می‌رسد. اما اگر طبیعت باز بودن پروژه را درست درک کرده باشید و متوجه شده باشید که هر کسی در این گونه همکاری به یک اندازه سود می‌برد، مسئله شراکت با رقبا را بهتر قبول خواهید کرد. البته همچنان باید با وکیل خود برای پرهیز از رفتارهای ضدانحصار (antitrust)، مشورت کنید؛ زیرا قانون، نوع همکاری با رقبا را محدود کرده است.

۱۱-۴- نگهداری

فاز نگهداری با عمومی شدن پروژه آغاز می‌شود. اکنون شما برنامه‌های بازاریابی خود را با موقیت اجرا کرده‌اید و یک جامعه‌ای از برنامه‌نویسان حول پروژه شما در حال شکل‌گیری است. این فاز را باید به دقت مدیریت کرد تا موقیت پروژه تضمین شود. در مورد اکثر پروژه‌ها باید فعالیتهای بازاریابی و آگاهی دادن به جامعه را همچنان ادامه دهیم.

ستجش

در فاز برنامه‌ریزی تجاری باید برخی از معیارهای تضمین کننده موقیت را تعیین کرده باشیم. تعداد برنامه‌نویسان، کارهای عرضه شده و میزان تقاضا برای نرمافزار برخی از این معیارها هستند. هر یک از این معیارها را باید در بازه زمانی مشخصی اندازه‌گیری کنید و سپس بر اساس نتیجه‌ای که از آن می‌گیرید، اقدام کنید. به عنوان مثال اگر تعداد برنامه‌نویسان کمتر از حد نیاز است. باید برنامه‌های بازاریابی خود را گسترش دهید. به طور کلی باید تمامی معیارها را به همین ترتیب زیر نظر داشته باشید و متناسب با آنها واکنش نشان دهید.

بازار در حال پیشرفت

نباید شرایطی برای خود ایجاد کنید که پس از آغاز پروژه قادر به سرمایه‌گذاری در زمینه تبلیغ آن نباشید. در صنعت، تحولات و وقایع نسبتاً زیادی اتفاق می‌افتد و پروژه شما می‌تواند از این وقایع استفاده کند. به عنوان مثال، فرض کنید پروژه‌ای در زمینه برنامه‌های تلفن همراه در دست اجرا دارید و شرکتی محصول یا استاندارد جدیدی در زمینه این صنعت ارائه می‌کند. شما می‌توانید با ارتباط دادن این اتفاق جدید به پروژه خود، برای آن تبلیغ کنید و آگاهی عمومی نسبت به آن را افزایش دهید.

برنامه‌نویسان تازه وارد، مدام در حال پیوستن به صنعت هستند. اطلاعات سال قبل برای این تازه واردها تازگی دارد. به روز کردن و آموزش پیوسته برنامه‌نویسان، باعث می‌شود که جامعه همواره در زمینه پروژه شما پویا باشد.

اهداف راهبری

یکی از جالبترین نکات در زمینه متن باز (که البته اکثر مدیران را نگران می کند) این است که جامعه برنامه نویسان پروژه شما را به جاهایی می برد که تصور نمی کردید. قسمتهایی از نرم افزار شما ممکن است در جاهایی استفاده شود که پیش بینی آن را نمی کردید. در اغلب موارد به این وقایع باید از دید مثبت نگاه کنیم. اینها علامت فعالیت و رشد جامعه برنامه نویسان است. در عین حال باید مراقب انحراف پروژه از اهداف پیش بینی شده آن باشیم.

در اینجا پویایی جالبی صورت می گیرد که نیازمند نظارت دقیق شماست. این امکان وجود دارد که جامعه متن باز بخواهد پروژه را در جهتی غیر از آنچه که شما در نظر گرفته اید هدایت کند. این جهت گیری ممکن است بهتر از آنچه باشد که برای آن در نظر گرفته بودید. در اینجا شما باید پیشنهادها را با ذهنی باز دریافت کنید و آنها را دقیقاً بررسی کنید. به هر حال این وظیفه شماست که خود را به عنوان راهبر پروژه بشناسانید و در زمینه اهداف آینده پروژه مذاکره کنید. در کل باید غرور را کنار گذارید تا فرصت هایی که از طریق پیشنهادهای ارائه شده، در اختیار شما قرار می گیرد از دست ندهید.

۲-۱۲- متن باز داخلی

به فرآیند به کارگیری نرم افزارهای موجود متن باز در محصولات یا پروژه های داخل سازمانی، «متن باز داخلی» گویند. قواعد و فرآیندهای مرتبط با متن باز داخلی با متن باز خارجی بسیار متفاوت است. در این فرآیند کسب و کار شما باید قوی باشد. در ادامه به برخی از انگیزه های متن باز داخلی اشاره می کنیم:

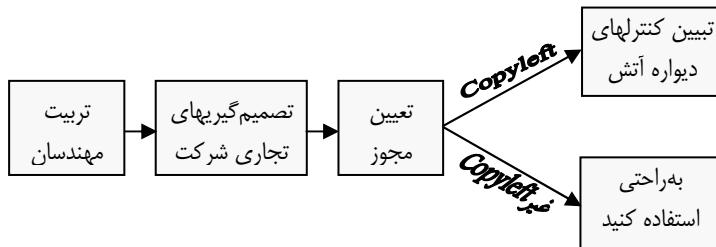
- استانداردها- اگر گونه متن بازی از استانداردهای مورد نیاز کسب و کار شما وجود داشته باشد، بهتر است محصول خود را بر اساس آن استاندارد تهیه کنید. مزیت این کار این است که لزومی ندارد خودتان مجدداً آن استاندارد را تهیه و سپس استفاده کنید.
- فناوری های موجود- اگر جامعه متن باز فناوری تهیه کرده باشد که صنعت آن را پذیرفته است، می توان با استفاده از آن فناوری سیر پیشرفت محصول خود را سریعتر کنید.
- منابع- اگر از فناوری های موجود در زیربنای محصول خود استفاده کنید، می توانید نیروی کار خود را بر بخش هایی متتمرکز کنید که درآمدزائی بیشتری دارد.
- خطر دوطرفه بودن (بازگرداندن کد)- مجوزهای GPL و تعدادی دیگر از مجوزهای متن باز شما را ملزم به ارائه تغییرات و اصلاحات برنامه به جامعه متن باز می کند. بنابراین باید در زمینه استفاده از این گونه کدها در محصول خود با دقت بیشتری تصمیم گیری نمایید.

به همین ترتیب شرایطی هم وجود دارد که استفاده از متن باز داخلی باید پرهیز شود.

- جهت گیری استراتژیک- گرچه پروژه های متن باز می توانند نیازهای فوری شما را برآورده کند، اما ممکن است با اهداف آینده شما سازگاری نداشته باشد. شما باید در زمینه اهداف آینده نگهدارنده پروژه متن باز مورد نظر مذاکره کنید و بدانید که پروژه

- وی در آینده به چه سمتی خواهد رفت. اگر اهداف آینده پروژه را نادیده بگیرید ممکن است مجبور شوید خودتان به تهایی نگهداری و پشتیبانی پروژه را به عهده گیرید.
- نیروی فنی- اگر راهبران کلیدی پروژه مخالف استفاده از متن باز باشند، حداقل کاری که باید انجام شود این است که طراحی آن بازنگری شود. در چنین شرایطی باید بین نیاز به فناوری جدید و مخالفتهای موجود تعادل برقرار کرد. در نهایت اول باید مطمئن شوید که تیم شما در جهت اهداف درآمدهای حرکت می‌کند و به تدریج گزینه‌های مربوط به متن باز را با سیاست به آنها تحمیل کنید.
 - کنترل عرضه محصول- همان گونه که می‌دانید جامعه متن باز وقتی نرم‌افزاری را عرضه می‌کند که نگهدارنده آن به این تشخیص برسد که زمان آن فرا رسیده است. اگر نیازمند کنترل دقیقی در زمان عرضه محصول هستید، ممکن است نتوانید منتظر بمانید تا جامعه اصلاحات مورد نظر شما را انجام دهد. البته در چنین شرایطی یک راه این است که با جامعه همکاری کنید و خودتان اصلاحات مورد نیاز را روی پروژه انجام دهید.

در شکل ۲-۱۲ فرآیندهای مورد نیاز برای داشتن کنترل قابل قبولی بر روی پروژه متن باز داخلی نمایش داده شده است.



شکل ۲-۱۲: فرآیند متن باز داخلی

همان گونه که می‌بینید اولین قدم، داشتن گروه مهندسی است که به خوبی تربیت شده باشد.

۲-۱-۱- تربیت مهندسان

برای یک مهندس، رفتن به اینترنت و پیدا کردن کدهای مفید و استفاده آنها در یک پروژه بسیار آسان است. مشکل اینجاست که اگر مهندسان خود را آموزش نداده باشید، ممکن است از کدهایی استفاده کنند (مثلاً کدهای دارای مجوز GPL) که وضعیت اختصاصی بودن محصول را به مخاطره بیندازد. عده‌ای فکر می‌کنند اگر از کدهای GPL به طور مخفیانه استفاده کنند، هیچ مشکلی برایشان پیش نخواهد آمد و اگر هم به دادگاه کشیده شوند پیروز خواهند شد. این تصور کاملاً غلط است. بلکه عواقب این کار این است که شما پس از پرداخت جریمه تعیین شده،

مجبور خواهید شد کدهای GPL را از محصول خود خارج کنید. به هر حال اگر مهندسان خود را آموزش دهید هزینهٔ شما بسیار کمتر خواهد بود.

در کل، باید مراقب باشید کاری نکنید که مجبور به پس گرفتن محصول خود از مشتریان شوید. این مسئله از در اختیار عموم قرار گرفتن دانش فنی سازمان مهمتر است. به همین دلیل شما باید مهندسان خود را به خوبی آموزش دهید تا از خطرات به کارگیری کدها از منابع نامشخص و تأثیر آن بر کسب و کار شما آگاه باشید.

۱۲-۲-۲- تصمیم‌گیریهای مربوط به سازمان

همچنان که رقابت شما با جامعهٔ متن‌باز بیشتر می‌شود، سازمان باید سیاستهایی نیز برای همراه شدن با آن داشته باشد. مسلماً این کتاب به این منظور نوشته شده است که اطلاعات کافی در زمینهٔ مزایای استفاده از نرم‌افزارهای متن‌باز در اختیار شما قرار دهد. با این حال حتی اگر شما سیاستی را دنبال کردید که کلاً متن‌باز را داخل سازمان ممنوع می‌کند، این نکته را نباید فراموش کنید که برای مهندسان استفاده از کدهای موجود در اینترنت بسیار آسان است. به همین دلیل، باید سیاستهای شما به طور واضح و روشن به آگاهی مهندسان برسد.

اگر قصد دارید استفاده از متن‌باز را توسط مهندسان آزاد گذارید، باید سیاستهای خاصی را دنبال کنید. این سیاستها باید موارد زیر را پوشش دهند:

- چه سطحی یا دسته‌ای از پروژه‌ها می‌توانند از نرم‌افزارهای متن‌باز استفاده کنند؟ مجوز یا مجوزهای این نرم‌افزارها چه باشد؟ مهندسان هنگام استفاده از منابع شرکت یا منابع موجود در اینترنت چگونه باید عمل کنند؟ این نرم‌افزار چگونه داخل سازمان مدیریت می‌شوند؟ نحوهٔ پشتیبانی نرم‌افزار عرضه شده که حاوی کدهای متن‌باز است، چگونه باشد؟

در فصل بعد مسائل مربوط به کارمندان را بیشتر بررسی خواهیم کرد که در تصمیم‌گیری بهتر به شما کمک خواهد کرد.

۱۲-۳-۲- تعیین نوع مجوز

همان گونه که می‌دانید دسته‌ای از مجوزها وجود دارد که به صورت دوطرفه (رفت و برگشتی) هستند. یعنی شما را ملزم به بازگرداندن تغییرات انجام شده روی نرم‌افزار می‌کند. GPL و LGPL متداول‌ترین این دسته مجوزها هستند. اگر از این گونه کدها استفاده می‌کنید، باید کنترل مهندسی بسیار دقیقی بر روی پروژه داشته باشید تا بتوانید از ترکیب کدهای متن‌باز و اختصاصی جلوگیری کنید.

اگر مجوز کدهای استفاده شده به صورت دوطرفه نباشد، مانند BSD یا MIT می‌توانید به راحتی از آنها در پروژه خود استفاده کنید. اما همچنان نباید کار با جامعهٔ متن‌باز را فراموش کنید. زیرا با این کار می‌توانید آنها را در زحمت پشتیبانی نرم‌افزار شریک کنید.

۱۲-۴- دیوار آتش

اگر شرکت شما هم محصولات اختصاصی و هم محصولات متن باز دارد، باید دیوار آتشی بین این دو محیط ایجاد کنید. بهترین راه ایجاد این دیواره این است که از دو سامانه مدیریت نرم افزار جداگانه استفاده کنید یا اینکه فقط از مخزن های عمومی متن باز شناخته شده، برای این گونه برنامه های خود به کار ببرید. وظایف مهندسان را نیز باید به درستی تعیین کنید. اگر مهندسی بر روی هر دو نوع نرم افزار کار می کند، باید قبلاً به خوبی تربیت شده باشد تا سه ها کدهای متن باز را وارد کدهای اختصاصی نکند، مگر اینکه مجوز آن کد، این کار را مجاز شمرده باشد. حتی در مواردی که تیمهای مهندسی متن باز و اختصاصی به طور مستقل کار می کنند، باز هم باید مراقب منبع کدها بود.

۱۲-۳- توسعه فناوری اطلاعات

اگر قصد دارید در نرم افزارهای داخلی خود از متن باز استفاده کنید، دیگر نباید نگران مجوز کدها باشید. تمامی مجوزهای متن باز این کار را مجاز می دانند. مگر اینکه در آینده تصمیم به عرضه نرم افزار بگیرید. در این صورت باید مراقب عواقب منفی احتمالی این کار باشید:

- اگر کدی را استفاده کنید و تغییرات آن را باز نگردانید از آن پس باید خودتان از نرم افزار [نگهداری و] پشتیبانی کنید.
 - اگر در هر زمان شرکت تصمیم به عرضه پروژه های داخلی خود کند، در صورتی که این پروژه ها وابسته به اجزاء متن بازی باشد که قابل تفکیک از بخش های اختصاصی نباشند، این کار غیرممکن خواهد بود.
- بهترین راه این است که همواره با جامعه متن باز همکاری کنید و همیشه در معماری نرم افزار، اجزاء متن باز و اختصاصی را به درستی تفکیک کنید.

۱۲-۴- جبران زیان

اگر نرم افزار تجاری شما از نرم افزار شرکت ثالثی استفاده می کند، در قراردادی که با آن شرکت می بندید، اغلب بندهایی برای جبران زیان در نظر گرفته می شود. یکی از دلایل وجود این بندها این است که اگر این نرم افزار شرکت ثالث از مالکیت معنوی شخص دیگری بدون مجوز استفاده کرده باشد، شما مতهم نشوید. طبیعت اکثر نرم افزارهای متن باز به گونه ای است که اجزای خود را از منابع مختلف جذب می کند که بسیاری از آنها قابل ردیابی نیستند.

حتی اگر یک شرکت بزرگ، نرم افزار متن بازی را به عنوان بخشی از نرم افزار تجاری خود به شما بفروشد، در صورت هر گونه شکایتی، آن شرکت نمی تواند پای شما را به میان بکشد. اصطلاح عدم جبران خسارت در صنعت نرم افزار به تدریج در حال جا افتادن است. عدم جبران خسارت بدین معناست که به عنوان مثال اگر نرم افزاری را به طور رایگان از اینترنت دریافت کردید، در اغلب موارد کسی نمی تواند از شما شکایت کند.

۵-۱۲- جمع‌بندی

در این فصل، زیربنای فکری مورد نیاز برای تصمیم‌گیری در زمینه استفاده از متن‌باز در داخل سازمان، در اختیار شما قرار گرفت. اکنون باید درک درستی از مفاهیم عرضه نرم‌افزار به جامعه متن‌باز و استفاده از کدهای متن‌باز در سازمان، پیدا کرده باشید. بنابراین شما تقریباً با تمامی اجزای لازم برای تعیین خطا مشیهای شرکت خود آشنا شده‌اید. آخرین بخشی از این اجزا که باید بررسی شود، منابع انسانی است. برخی از اجزا منابع انسانی به شما کمک خواهد کرد که سیاست متن‌باز شرکت خود را تکمیل کنید و سایر اجزا آن، به شما در استخدام تیم خبره و همکاری مؤثر با جامعه متن‌باز کمک می‌کند.

فصل سیزدهم

منابع انسانی: استخدام بهترین استعدادها

در بسیاری از شرکت‌ها در سیاستهای استخدامی تصريح می‌شود، هر کاری که کارمندان انجام می‌دهند متعلق به شرکت است. اما اگر کارمندی بر روی پروژه‌ای متن‌باز کار کند، تکلیف چیست؟ چگونه ممکن است کارمندی بتواند برنامه متن‌باز خود را با توجه به این شرایط، به جامعه متن‌باز برگرداند؟ متن‌باز نحوه نگاه به منابع انسانی و ساختار سازمانی را از پایه تغییر داده است. در این فصل ما جزئیات فرآیند استخدام و نحوه ارزیابی کارمندان را بررسی خواهیم کرد. همچنین به برخی از موارد لازم برای تأمین مالکیت معنوی در فرآیند استخدام خواهیم پرداخت. همچنین خواهیم دید چگونه می‌توان جامعه متن‌باز را جزئی از ساختار سازمانی تلقی کرد. در این فصل نحوه مدیریت دیواره‌های آتش را نیز بررسی خواهیم کرد. هدف این فصل این است که نحوه انجام موارد زیر را بیاموزیم:

- تکامل فرآیند استخدام
- مدیریت مالکیت معنوی کارمندان، شرکت و جامعه متن‌باز
- همکاری با رهبران جامعه متن‌باز

استخدام و کار با برنامه‌نویسان جامعه متن‌باز، فرآیند دیگری است که به شما در شناخت افرادی که استخدام می‌کنید و تعیین وظیفه اولیه آنها کمک می‌کند.

۱۳-۱- قراردادهای استخدامی

در استخدام مهندسانی که قبلًا در جامعه لینوکس و متن‌باز کار می‌کردند و تعصب عمیقی نسبت به آن دارند، به مسایل جدیدی برمی‌خورید. چنین اشخاصی از اینکه در شرکتی استخدام می‌شوند که بر روی پروژه‌های متن‌باز کار می‌کند، بسیار هیجان‌زده خواهد شد. با این حال ممکن است انتظاراتی از قبیل اینکه فقط بر روی پروژه‌های متن‌باز نیاز داشته باشد. بنابراین لازم است شرکت سیاستهای جدیدی برای استخدام این افراد تعیین کند. در این سیاستها باید موارد زیر در نظر گرفته شود:

- پروژه‌ها- آیا کارمندی می‌تواند کار خود را محدود به پروژه‌های متن‌باز کند؟ برای شرکتهای بزرگی که سابقه زیادی در همکاری با جامعه متن‌باز دارند، این کار اشکالی ندارد. اما اگر قصد دارید که کارمندان را در پروژه‌های اختصاصی نیز درگیر کنید، این سیاست شما باید به وضوح به اطلاع آنها برسد.
- مالکیت حق کپی- همان گونه که در این کتاب یاد گرفتید، حتی حق کپی کدهای متن‌باز نیز مالکیت دارد و محفوظ است. در سیاستهای شما باید مشخص شده باشد که مالکیت حق کپی کارهای انجام شده توسط مهندسان، متعلق به کیست. آیا متعلق به شرکت است یا متعلق به کارمند یا هر دو؟ در اغلب موارد شرکت باید مالکیت حق کپی کارها را داشته باشد تا بتواند در صورت نیاز از سیاست مجوز دوگانه استفاده کند. اما باید در نظر داشت در مواردی نظیر آپاچی و سامبا مجوز آنها به گونه‌ای تعیین شده

است که امکان اعطای حق کپی کارهای انجام شده بر روی آنها به شخص دیگری وجود ندارد.

- وقت آزاد مهندسان و وقت کار- بسیاری از مهندسان در وقت آزاد خود بر روی پروژه‌های متن باز دیگر کار می‌کنند. ممکن است مرز بین کار شرکت و سرگرمی کارمند مشخص باشد یا نباشد. در عین حال ممکن است کاری را که یک کارمند به صورت سرگرمی آغاز کرده است، بتوان تبدیل به پروژه شرکتی کرد. به هر حال باید نحوه کار کارمندان بر روی پروژه‌های متن باز در وقت آزاد آنها را مشخص کنید و قوانینی برای آن وضع کنید. درنظر داشته باشید در برخی مناطق، ممکن است قوانین محلی مانع محدود کردن کارمندان شود.
- رقابت- سیاست منابع انسانی شما احتمالاً امکان کار کارمندان با رقبا را نفی می‌کند. یکی از مزایای متن باز این است که می‌توان رقبا را در هزینه‌های پروژه شریک کرد (با در نظر گرفتن محدودیتهای قانون ضد انصار). در چنین مواردی مهندسان شما می‌توانند فعالانه در پروژه‌های رقبای شما نیز کار کنند. اگر مجموعه سیاستها و راهکارهای شما این موارد را پوشش داده باشد، شما باید بتوانید به راحتی از استعدادهای موجود در پروژه‌های متن باز خود استفاده کنید.

۲-۱۳- سیاستهای کار در پروژه

برای کارمندان خود اهداف کسب و کار خود را مشخص کنید. لازم است به برخی از کارمندان یادآوری کرد که کسب و کار شما باید سودآوری داشته باشد. حتی اگر بر روی پروژه‌های کار می‌کنید که مستقیماً درآمده نیست، تبیم آن پروژه باید بداند که چگونه کسب درآمد کند (مثلاً پشتیبانی، خدمات، سخت‌افزار و غیره). اگر مدل تجاری خود را به مهندسان توضیح دهید، آنها خواهند توانست در حالی که به طور مؤثر با جامعه متن باز کار می‌کنند، بین نرم‌افزارهای اختصاصی و متن باز مرز قائل شوند و بهتر تضمیم‌گیری کنند. همچنین باید نحوه رفتار خود با جامعه متن باز را معین کنید. از چه مجوزهایی برای پروژه‌های خود استفاده کنید تا امکان کار کارمندان بر روی آن را بدهد؟ اگر بر روی پروژه‌های هسته لینوکس کار می‌کنید، سیاست شما در زمینه پیمانه‌های اختصاصی و متن باز هسته چیست؟ به طور خلاصه شما باید عناصر بررسی شده در این فصل را، کنار عناصری که در فصلهای قبل یاد گرفتید بگذارید تا بتوانید یک سیاست جامع متن باز برای شرکت خود تهیه کنید.

۳-۱۳- استخدام شخص مناسب

در فرآیند استخدام باید بتوانید تشخیص دهید که آیا شخص جدید مناسب سازمان شما هست یا نه؛ باید از معیارهایی که از قبل تعیین کرده‌اید به علاوه برخی معیارهای دیگر ذکر شده در این قسمت، برای تشخیص مناسب بودن یک کارمند استفاده کنید. این معیارهای اضافه را در ادامه بررسی می‌کنیم:

۱۳-۳-۱- فناوری

هر فناوری، (هسته لینوکس، سامانه پرونده، کارساز وب، راه اندازها، پشتیهای شبکه و غیره)، یک جامعه برنامه نویس فرعی خاص خود دارد. کارمندی که استخدام می کنید، گرچه ممکن است در جامعه بزرگ متن باز شناخته شده نباشد، اما ممکن است در جامعه فرعی تخصصی خود مشهور باشد. بنابراین دنبال شهرت وی در جامعه لینوکس و متن باز نباشید، بلکه به سراغ جامعه های فرعی بروید و میزان شهرت او را در آنجا جویا شوید. برای این کار کافیست با اعضای جامعه فرعی مورد نظر تماس بگیرید و نام افراد کلیدی آن جامعه را پرسید.

۱۳-۳-۲- خانه جامعه

هر پروژه ای یک نگهدارنده دارد. وظیفه نگهدارنده این است که کارهای عرضه شده را قبول یا رد کند و در زمینه زمان مناسب برای عرضه نرم افزار، تصمیم گیری کند. گاهی جوامع جدید در نتیجه عدم توافق بین یک برنامه نویس و نگهدارنده نرم افزار، شکل می گیرد. به این اتفاق «منشعب شدن^۱» گویند که البته به ندرت اتفاق می افتد. این از خوشایندترین فرآیندها در متن باز است. اما در مواردی این کار به صرفه نیست (مثلاً در زمینه هسته لینوکس). نگهدارنده های خوب آنها هستند که سابقه طولانی در حل مشکلات فناوری خاص دارند و تمامی انرژی خود را بر روی یکی از پروژه های آن فناوری متمرکز کرده اند. شما می توانید برای یک پروژه خاص یک نگهدارنده استخدام کنید. در چنین شرایطی باید تعادلی بین وظایف او نسبت به شرکت و نسبت به جامعه متن باز برقرار کنید. آیا شرکت می تواند با مواردی که کارمندی تصمیم گیری می کند که به ضرر شرکت تمام می شود، برخورد کند؟ فکر نکنید صرف اینکه نگهدارنده را استخدام کرده اید، پروژه را تحت کنترل خود دارید. به یاد بیاورید که لینوس توروالز نگهدارنده هسته لینوکس است نه شرکت ترانس متا، که در آنجا کار می کند.

۱۳-۳-۳- نگهدارنده یا برنامه نویسان

همان گونه که تاکنون متوجه شده اید، اکثر پروژه های متن باز با نگهدارنده و جامعه برنامه نویسان آنها شناخته می شوند. شما باید به خوبی بدانید که چه نقشی را در اینجا می خواهید بر عهده بگیرید. برای موفقیت باید ذخیره قوی ای از برنامه نویسان به همراه تعداد زیادی راهبر نزد خود داشته باشید. اگر بتوانید از افرادی استفاده کنید که در زمینه فناوری های مختلف فعالیت کرده اند، برد خوبی خواهید کرد. این افراد درک خوبی از نحوه کار جوامع مختلف دارند و با نگهدارنده های پروژه های مختلف کار کرده اند. شما می توانید برای شناخت ارزش یک برنامه نویس با نگهدارنده پروژه هایی که او در آنها دخیل بوده است، مشورت کنید.

اگر نیازمند یک راهبر برای پروژه ای هستید، می توانید از افرادی که تجربه نگهداری پروژه را دارند استفاده کنید. برای محک زدن یک نگهدارنده می توانید از موارد زیر استفاده کنید:

- آیا او برنامه نویسان کلیدی را می شناسد و از آنها بخوبی استفاده می کند؟

¹ Forking

- چگونه اختلاف و تداخل بین برنامه‌نویسان را حل می‌کند؟
- آیا او با پروژه‌های رقیبی که در نتیجه عدم توافق ایجاد شده‌اند، آشناست؟ آیا می‌شد این مشکلات را به نحوه بهتری حل کرد؟
- آیا او قادر است برنامه‌نویسان کلیدی را جذب پروژه خود کند؟
- آیا او درگیر فرآیندهای چرخه عرضه نرم‌افزار بوده است؟
- چگونه تشخیص می‌دهد که زمان عرضه نرم‌افزار فرا رسیده است؟
- آیا او می‌تواند کدهایی را به شما نشان دهد که بهتر از کارهای خودش هستند؟ در حقیقت، نگهدارنده‌ها افرادی هستند که می‌توانند با استفاده از مهارت‌های فنی خود و تعامل مؤثر با برنامه‌نویسان، روند توسعه یک پروژه را هدایت کنند.

۱۳-۴-۳- شهرت و احترام در جامعه

کسی می‌تواند یک پروژه را در جامعه راهبری کند که فعال باشد و کارهای خوبی برای پروژه ارائه کرده باشد. کنترل پروژه بر عهده نگهدارنده است. نگهدارنده پروژه نحوه کار افراد و کیفیت آنها را می‌داند. آیا کارمندی را که می‌خواهید استخدام کنید در جامعه متن باز از شهرت و احترام برخوردار است؟ برای تشخیص میزان شهرت و احترام او، می‌توانید به لیست‌های پستی و آرشیوهای موجود از پیامهای رد و بدل شده میان برنامه‌نویسان مراجعه کنید.

۱۳-۵-۳- تعامل اینترنتی

وقتی که از جوامعی که کارمند در حال استخدام، در آنها درگیر بوده است آگاه شدید و به این نتیجه رسیدید که او نیازهای شما را برآورده می‌کند، زمان آن می‌رسد که نحوه تعامل او با دیگران را بررسی کنید. تعامل از طریق اینترنت، زیربنای مهارت روابط عمومی افراد است. این مسئله بسیار حائز اهمیت است که او بتواند بر سایر افراد، در مناطق دیگر دنیا، تأثیر گذارد. برای فهمیدن این موضوع می‌توانید از موارد زیر استفاده کنید:

- به آرشیو پستها در لیستهای پستی الکترونیک مراجعه کنید و میزان تعامل وی را بررسی کنید. در نحوه تعامل وی دقیق شوید. آیا او با فرهنگ شرکت شما سازگار است؟ آیا می‌تواند در بهبود کارهای عرضه شده توسط کارمندان مؤثر باشد؟
 - ببینید او از چه نامهای مستعاری برای خود استفاده کرده است. آرشیوهای SlashDot (به بخش مرجع کتاب مراجعه کنید) و سایر آرشیوهای محلی را بررسی کنید. آیا او با استفاده از نام مستعار فریبیه کاری می‌کند؟ آیا او احساسی عمل می‌کند یا منطقی؟
 - آیا او در زمینه موضوعات مفید آغاز بحث می‌کند یا اینکه فقط به دیگران پاسخ می‌دهد؟ آیا پاسخهای او مفید هستند؟
- سعی کنید میزان احترام او را از طریق تعداد افرادی که دنبال راهنمایی‌های وی هستند، تشخیص دهید. همچنین میزان پذیرش کارهای او و توسط نگهدارندها را بررسی کنید. کسی که در جامعه دارای احترام باشد، کارهایش بدون هیچ بحثی پذیرفته می‌شود.

۱۳-۳-۶- کارهای عرضه شده

از او بخواهید که لیست کاملی از کارهایی که در پروژه‌های مختلف متن باز انجام داده است، در اختیار شما قرار دهد. اگر او در پروژه اختصاصی شرکت دیگری کار کرده باشد، اغلب نمی‌توان از کیفیت و ارزش دقیق این کار وی مطلع شد. زیبایی متن باز در اینجاست که همه چیز در اختیار عموم است. شما می‌توانید به راحتی کیفیت، کمیت و توانایی فنی مقاضی استخدام را بستجید. می‌توانید از مدیران و سایر کارمندان خود برای تحلیل کارهایی که او انجام داده است، طلب کمک کنید.

همچنین شما باید سبک کارهای او را نیز بررسی کنید. آیا او فقط مشکلات خاص را حل می‌کند یا در مباحث طراحی و معماری پروژه نیز دخیل است؟ سبک کارهای او باید با نیاز شما همخوانی داشته باشد. به عنوان مثال اگر نیازمند شخصی برای پشتیبانی هستید، شخصی که در زمینه حل مشکلات کار کرده است، مناسب این موقعیت می‌باشد. اما اگر به دنبال یک نگهدارنده و راهبر هستید، باید کسی را پیدا کنید که سابقه درخشنانی در برنامه‌نویسی و مدیریت کارهای سنگین داشته است.

۱۳-۳-۷- جغرافیا

هر کس که در زمینه کار با جامعه متن باز تجربه داشته باشد، قادر است با سایر افراد از مناطق و فرهنگها مختلف دنیا تعامل کند. وقتی که قصد استخدام شخصی را دارید، باید تعیین کنید که مقاضیان از چه مناطقی می‌توانند باشند و پس از استخدام در کجا ساکن خواهند شد.

اینکه بتوانید افراد مورد نیاز را از هر جای دنیا استخدام کنید، مزیت بزرگی برای شرکت شمامست. اما داشتن تیمی که از نظر مکانی در کنار یکدیگر کار می‌کنند نیز دارای مزایای زیادی است. بر اساس تجربه من، یکی از بهترین روشها این است که تیم هر پروژه، در یک نقطه مرکز باشد. از پخش کردن پروژه‌ها در مناطق مختلف پرهیز کنید. شما می‌توانید بعدها، در هر منطقه تیم ماهری برای بومی‌سازی پروژه و پشتیبانی آن تعیین کنید. مزیت دیگر این روش این است که تیم پروژه می‌تواند با برنامه‌نویسان محلی همانند افراد جامعه متن باز ارتباط برقرار کند.

اگر شما بر روی پروژه متن بازی کار می‌کنید که برای شرکت بسیار حائز اهمیت است، سعی کنید جلسات دوره‌ای با افراد کلیدی جامعه متن باز داشته باشید. به این گونه جلسات معمولاً «جشن هک»^۱ می‌گویند. اگر بتوانید این جلسه را نزدیک یکی از واقعی (سمینارها، کنفرانسها) مربوط به نرم‌افزار یا کسب و کار خود برگزار کنید، می‌توانید جمع کثیری از برنامه‌نویسان را یکجا جمع کنید تا با کسب و کار و پروژه شما بیشتر آشنا شوند. سعی کنید تمامی هزینه‌های آنها را، از قبیل بلیط هوایپیما، هتل و غذا، پردازید. شما متوجه خواهید شد که شرکت‌کنندگان در این جلسات بسیار قانع هستند و بیشتر دوست دارند برنامه‌نویسی کنند تا اینکه پول شما را به هدر دهند. هدف شما باید این باشد که هر آنچه که برنامه‌نویسان لازم دارند در اختیار آنها قرار دهید

¹ Hack-Fest

و یک چرخه تولید جدید را آغاز کنید. به این ترتیب از همکاری این افراد نتیجه بسیار خوبی خواهد گرفت.

۸-۳-۱۳- مراحل را بشمارید

یک جامعه عمده‌اً از طریق انگیزه‌هایش حرکت می‌کند. مهمترین کسی که می‌تواند در افراد پروژه انگیزه ایجاد کند، نگهدارنده آن است. در زمینه پروژه‌های بزرگ سلسله مراتبی از اعتماد به طور غیررسمی بین برنامه‌نویسان و نگهدارنده وجود دارد. مسلماً نباید انتظار داشت که همیشه به راحتی می‌توان نگهدارنده یک پروژه دیگر و برنامه‌نویسان او را به استخدام خود درآورد. در جریان استخدام یک برنامه‌نویس، باید این سلسله مراتب را کشف کنید و تعداد برنامه‌نویسان بین او و نگهدارنده را بشمارید. به این کار «شمردن واسطهها»^۱ می‌گویند؛ همان گونه که در مسیردهی ترافیک شبکه انجام می‌شود. هر چه مراحل کمتر باشد، متقارضی استخدام به نگهدارنده نزدیک‌تر است.

۴-۱۳- ساختاربندی تیمها

ساختاربندی تیمها بستگی به نقش شرکت شما در یک پروژه متن‌باز دارد. همان گونه که در مفهوم بازار شرکتی یاد گرفتید، شما می‌توانید یک پروژه متن‌باز بسیار بزرگ را داخل شرکت ساختاربندی کنید و سپس تیم حاصله را مدیریت کنید. اما اغلب شما بر روی پروژه‌های کوچک کار می‌کنید و معمولاً شما یکی از ارائه کنندگان کد به پروژه هستید. به خاطر مسئله دوطرفه بودن (بازگرداندن کد) که در فصل قبلی به آن پرداخته شد، شما باید تمام تلاش خود را بکنید تا برنامه‌نویسان برای پروژه‌های متن‌باز خارج سازمان نیز کار کنند. مورد خاص جوامع دروازه‌ای نیز در ساختاربندی تیمها نقش دارد. ممکن است شما درگیر ترکیبی از پروژه‌های مختلف از انواع اختصاصی، متن‌باز و پروژه‌های جوامع دروازه‌ای باشید. این وظیفه شماست که فرآیندهای خاصی را برای روشن کردن مالکیت معنوی کارها و مرز بین آنها تعییه کرده باشید.

۵-۱۳- استخدام راهبران بر جسته

در اینجا من یکی از تجربه‌های شخصی خود را در زمینه HP بازگو می‌کنم. به عقیده بسیاری، استخدام بروس پرنز، یکی از برجهسته‌ترین رهبران جامعه متن‌باز، کار خطرناکی بود که من انجام دادم. اولویت اول بروس این بود که نماینده جامعه متن‌باز باشد و جنبش متن‌باز را ترقی دهد. او در حال حاضر با تعدادی از تیمهای داخلی HP کار می‌کند و آنها را در زمینه کار با جامعه متن‌باز آموزش می‌دهد. گرچه بروس HP را تبلیغ و معرفی می‌کند، در موارد بسیار نادری، نظرات او با نظرات شرکت متفاوت است. من و بروس چند قانون برای رفع چنین مواردی وضع کرده‌ایم:

- بروس حق ندارد به HP و مدیران آن حمله کند.
- بروس حق ندارد اطلاعات محصولات اختصاصی را در اختیار عموم قرار دهد.

¹ Counting the Hops

- بروس باید مشخص کند که چه موقع نظر شخصی خود را اعلام می‌کند و چه موقع به نمایندگی از HP صحبت می‌کند.

این وظیفه من بود که با صنعت، مطبوعات، تحلیل‌گران، مشتریان، شرکاء و رقبا کار کنم و از اینکه آنها از نقش بروس در HP آگاهند مطمئن شوم. به نظر من گرچه نقش دوگانه بروس همچنان مشکلاتی را برای برخی از تیمهای مدیریتی ایجاد می‌کند، اما برای کسب و کار HP در زمینه متن‌باز و لینوکس بسیار مفید بوده است. بسیاری از افراد در این صنعت توانسته‌اند به خوبی از نقش دوگانه راهبران بهره گیرند. بروس و HP توانسته‌اند اعتبار خوبی برای یکدیگر کسب کنند.

راهبران متن‌باز اول باید جامعه خود را معرفی کنند، سپس شرکت را. در ابتدا پذیرفتن چنین اصلی برای شما مشکل است، اما اگر مزهای روشنی برای نحوه ارتباطات مشخص کنید، موقفيت‌آمیز خواهد بود. کارمندان شما باید نقش منحصر به فرد چنین اشخاصی را درک کنند و بدانند قوانینی که برای این اشخاص برقرار است، برای سایر کارمندان صدق نمی‌کند. سعی کنید رابطه‌ای مبتنی بر اعتماد با تمامی راهبرانی که قصد استخدام آنها را دارید برقرار کنید.

۱۳-۶- جمع‌بندی

ساختمانی، شکل‌دهی و مدیریت تیمهای مهندسی که پژوهه‌های متن‌باز و غیرمتن‌باز را ترکیب می‌کند بسیار متفاوت است. از نظر فرهنگی مهندسان شما بین وفاداری به جامعه یا شرکت به مشکل بر می‌خورند. شما باید از مشکل وفاداری دوگانه میان برنامه‌نویسان خود آگاه باشید. کارمندان به تدریج با آن کنار می‌آیند و دیگر هراسی از آن نخواهند داشت. در این فصل با معیارهای جدیدی برای استخدام افراد آشنا شدید. همچنین مواردی که باید در سیاست‌گذاریهای برنامه‌نویسی شرکت در نظر گرفته شود، مورد بحث قرار گرفت.