



Open Source Programování

<http://rtime.felk.cvut.cz/osp/>

Pavel Píša

<pisa@fel.cvut.cz>

<http://cmp.felk.cvut.cz/~pisa>

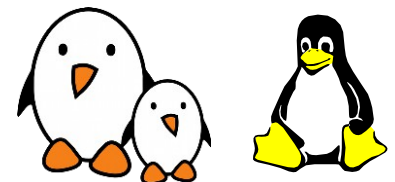
Michal Sojka

František Vacek

DCE FEL ČVUT



© Copyright 2004-2010, Pavel Píša, Michal Sojka, František Vacek,
Free-Electrons.com, GNU.org, kernel.org, Wikipedia.org
Creative Commons BY-SA 3.0 license Latest update: 19. II. 2014





Open source popisuje způsoby výroby a vývoje, které upřednostňují přístup ke kompletnímu výrobnímu postupu (zdrojovému kódu) pro všechny v procesu výroby, distribuce a užití zúčastněné strany.

Některými je viděn jako filozofie, jiní ho považují za pragmatický návod.

Termín je především svázaný se vznikem a rozšířením světové sítě Internet, které vyžadovaly tvorbu a vývoj nepřeborného množství kódu a jeho přizpůsobení nepřebornému množství prostředí.

- ▶ Že open source je ideální prostředí k pochopení, zkoumání jak a z čeho se SW vybavení a aplikace skládají
- ▶ Že open source je zajímavá volba pro každého
- ▶ Jak se organizuje vývoj malých i velkých softwarových projektů
- ▶ Jak se lze vypořádat se správou zdrojových kódů, řešením chyb, komunikací s uživateli
- ▶ Jak může open-source pomoci vám a jak můžete být prospěšní vy
- ▶ Jak komunikovat a jaká pravidla dodržovat, aby jste byli pochopeni, měli z práce radost a aby měli ostatní radost z vašich příspěvků
- ▶ Že na svobodě záleží (alespoň podle nás a našich zkušeností)

- ▶ V Open source moři existují projekty malé i velké, od firmware MP3 (RockBox), přes drivery, jádra operačních systémů (BSD, Linux, HURD, L4, Reactos), implementace knihoven (NewLib, GNU LibC), API (BSD sockety), ABI (Wine), překladače (GCC, Mono), GUI (Qt, Gtk, Fltk), uživatelská prostředí (KDE, GNOME), až po webové aplikace (Zope, MediaWiki) a projekty Wikipedia, OpenClipParts, OpenStreetMap
- ▶ Vše lze zkoumat, sledovat a modifikovat funkci, hledat návaznosti, kombinovat, upravit pro svojí potřebu a užitečné změny a znalosti lze sdílet s ostatními
- ▶ Přitom získané znalosti smíte použít v jiných aplikacích a to i uzavřených a nehrozí vám žaloby za vyzrazení tajemství, mnohaleté závazky mlčenlivosti a mnohaleté zakázky práce v určitém oboru
- ▶ Software můžete kopírovat, používat doma i v práci, nezavazujete se k EULA, nehrozí vám BSA

- ▶ Pro studium a to i pro ty, co jsou přesvědčeni o nadřazenosti projektů vedených na čistě uzavřené komerční bázi
- ▶ Pro užívání doma i v práci a to i přímo veškerý SW, se kterým se během předmětu setkáte – nic není zamknuté, nedostupné atd.
- ▶ Jedná se o neomezené hřiště, což je výzva pro každého kreativního jedince, pro některé i díky svobodě životní styl
- ▶ Pro toho, kdo chce především vydělávat, nabízí open-source také řadu možností – lze vydělávat na podpoře, správě, distribuci, tvorbě rozšíření
- ▶ Pro velké firmy je to jedna z mála možností jak spolupracovat bez úzkostného strachu z konkurence, patentů, právníků
- ▶ Možnost, jak prezentovat své schopnosti a získat i zaměstnání v open source nakloněných i svobodě vysloveně nepřátelských firmách

1. Úvodní slovo o předmětu; Open source software, projekt GNU, licence a operační systémy vycházející z jeho filozofie
2. Přehled významných Open source projektů (i pro výběr semestrální práce)
3. Verzovací systém Git; GNU - vznik základních vývojových nástrojů a jejich použití, křížový překlad
4. Linuxové jádro - vznik, vývoj, skladba a ovladače; GNU libc a uživatelský prostor
5. Grafická uživatelská prostředí a knihovny: QT, GTK+, GNOME, KDE
6. Komerční model, sociální a rozhodovací struktury; Souborové systémy, správa paměti, bloková zařízení.
7. Založení vlastního projektu a zapojení se do existujícího projektu; Technická infrastruktura (správa verzí a chyb, komunikace, wiki); Virtualizace
8. Mezilidská komunikace, dobrovolníci, vývoj a větvení projektů; Zpracování událostí (hlavní smyčka, thread pools, C10k)
9. Linuxové distribuce, vydávání verzí, příprava balíčků a průběžný vývoj.
10. Licence, autorská práva a patenty; Sledování živého systému, ladění, opravy a analýzy.
11. Cílové platformy, vestavěná zařízení, open source v průmyslu, přenositelnost a open source hardware
12. Trendy a výhled do budoucnosti

1. Seznámení se s předmětem a hodnocením, kontakt s reálným OS projektem (MC)
2. Tvorba "produktu" z nezávislých OS komponent ("embedded" Linux + BusyBox + jednoduchý modul do jádra)
3. Samostatná práce - specifikace individuální zadání a jeho zařazení do existujících OS projektů (konzultace s cvičícími)
4. Cvičení na vývoj a údržbu SW v GIT repository
5. Prezentace projektu do kterého budete přispívat a nástin řešení.
6. Písemka na znalosti z přednášek a práci s GIT repositářem
7. Samostatná práce
8. Samostatná práce
9. Samostatná práce
10. Samostatná práce
11. Samostatná práce
12. Prezentace (povinná účast)
13. Zápočet (povinná účast)

Co	Započet	Body	Pozn
1. cvičení (úprava MC)	*	5	a)
2. cvičení (dmsg vypise Hello <my name>)	*	5	a)
3. cvičení (předložení návrhu samostatné práce)	*	5	a)
4. cvičení (úprava patche proti Git historii)	*	5	a)
Prezentace záměru práce	*	5	
Test v půli semestru	*	15	
Odezva od člena projektu		3	
Vaše změny (i nepřijaté) jsou dostupné ve veřejném repozitáři		3	
Otestování vašich změn členem projektu		3	
Zahrnutí vaší implementace do projektu		3	
Vaše změny jsou zdokumentovány v uživatelské dokumentaci		3	
Výsledná práce odpovídá zadání ze 3. týdnu		3	
Profilová stránka a slidy k závěrečné prezentaci jsou anglicky		2	
Prezentace výsledku práce	*	10	b)
Ohloh KudoRank		5	c)
Zkouška		30	
Celkem		105	

Connect to people through the software you create & use
<http://www.ohloh.net/accounts/wentasah>



wentasah

SUMMARY

Profile

Journal

News

Kudos

Widgets

Stacks

My Projects

Experience

OCERA — Apr 2004 to Present

Developer at CZECH TECHNICAL UNIVERSITY IN PRAGUE

54 COMMITS



Kudos



Kudos Received

No kudos received yet.

Kudos Given



About Me

Homepage

<http://rtime.felk.cvut.cz/~sojka/>

Where



Praha, Česká republika

- ▶ Širokou nabídku OSS projektů, včetně konkrétních úloh, které mají zájem o příspěvky studentů představuje Google summer of code.
- ▶ Většina technologií společnosti Google je postavena na OSS a proto společnost sponzoruje rozvoj těchto projektů a platí studenty na nich pracující.
- ▶ Organizace zažádají o účast do 14. února. Schválené organizace do 24. února. Studenti diskutují o zadáních s organizacemi.
- ▶ Studenti mohou zasílat přihlášku od 10. března do 21. března. Organizace zažádá o podporu na konkrétní projekty do 7. dubna. Rozhodnutí o alokaci „stipendia“ 9. dubna.
- ▶ Studenti přijatí do GSOC získají všechny body za průběh samostatné práce. Povinnost závěrečné prezentace a její bodové hodnocení však zůstává.

- ▶ Analytické stroje pro výpočty (astronomie, kalendáře, obchod) a automaty pro (sekvenční) řízení v čase (z počátku spíše pro pobavení – hrací stroje) nejsou ničím novým
- ▶ Antikythera (pravděpodobně řecký ostrov Rhodes, Hipparchus 190 – 120 před n.l.), více jak 30 ozubených kol, 365.2422 dnů v roce, velmi přesný nelineární pohyb měsíce
- ▶ Pražský Orloj (Mikuláš z Kadaně a Jan Šindel – později profesor matematiky a astronomie na Karlově univerzitě 1410), 1552 Jan Táborský - oprava a spis o Hanušovi, postavy přidáné v 17. stol.
- ▶ Analytical Engine (Charles Babbage 1830)
- ▶ Projekt ENIAC (1944)
- ▶ Von Neumannova architektura ORDVAC (U-Illinois 1951)



- ▶ Antikythera byla pravděpodobně pravý open source - krycí desky obsahovaly více jak 2000 znaků dlouhý manuál, pravděpodobně včetně dokumentace použitých výpočtů pro neznalou osobu
- ▶ Naopak kdyby Hanuš poskytl veškeré know-how, tak by se nemohl pomstít a Orloj zastavit tak jak je to v pověsti



- ▶ Obecně rozvoji vědy a šíření vědomostí napomáhá otevřenost
- ▶ Prodeji jednotlivých výrobků naopak zisky zvyšuje jejich nedostatek a omezení konkurenční výroby (původně královské patenty, monopoly)
- ▶ Programy jsou pak specifickým materiálem, jejich návrh spíše odpovídá postupům matematickým a cena na vlastní fyzické šíření/kopírování je zanedbatelná. Vývoj a údržba jsou však velice nákladné.
- ▶ Je tedy programování hra, zábava, věda, řemeslná práce, výroba?

- ▶ 196x - MIT, AT&T Bell Labs, GE vyvíjeli inovativní systém Multics – projekt však byl příliš složitý a nezvládnutý
- ▶ Ken Thompson, Dennis Ritchie, M. D. McIlroy, and J. F. Ossanna se rozhodli, že zkusí napsat něco jednoduššího sami, aby si mohli pouštět hru Kena Thompsona Space Travel, i po tom, až bude MULTICS zrušen
- ▶ Použili nevyužitý PDP-7, napsali hierarchický souborový systém, správu procesů a zařízení, interpret příkazů a pár pomocných programků
- ▶ 1970 - Brian Kernighan pojmenoval systém Unics (Uniplexed Information and Computing Service)
- ▶ Až do té doby nedostali na projekt žádné prostředky, za slib tvorby utilit pro práci s texty (pro patentové oddělení) dostali PDP-11/20
- ▶ 1973 - Unix byl přepsán do jazyka C (Dennis Ritchie) a tím tak vyvrátil přesvědčení, že na systémové úrovni lze použít pouze assembler



1969

A jak to bylo dál



Legend:

- Open Source (Green box)
- Mixed/Shared (Orange box)
- Closed Source (Red box)

1971 to 1973

1974 to 1975

1978

1979

1980

1981

1982

1983

1984

1985

1986

1987

1988

1989

1990

1991

1992

1993

1994

1995

1996

1997

1998

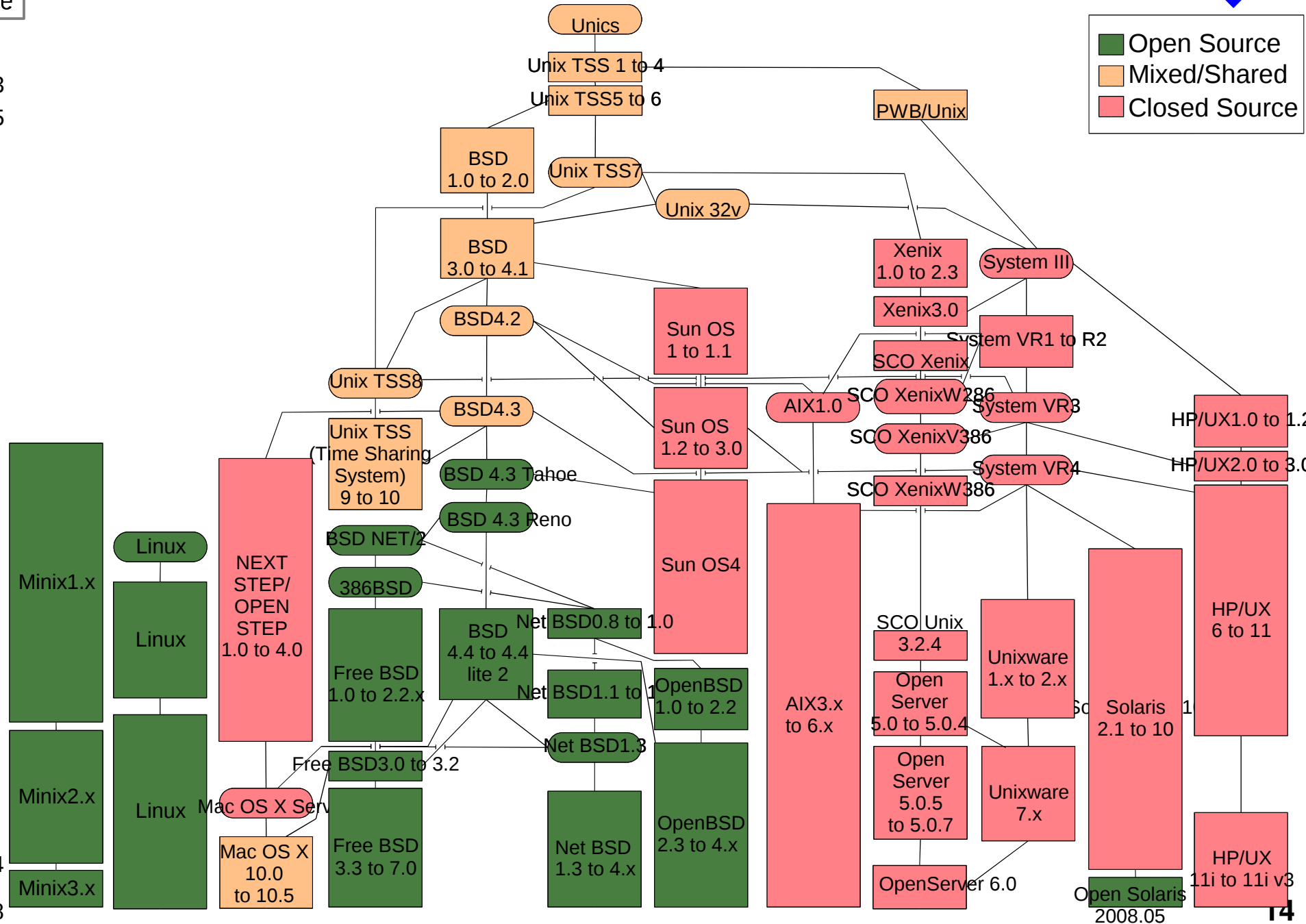
1999

2000

2001 to 2004

2005

2006 to 2008



- ▶ 1971 - MIT Artificial Intelligence Lab, Richard M. Stallman (RMS)
- ▶ Sdílení SW, komunita, přístup stejný jako výměna receptů
- ▶ Digital PDP-10, Incompatible Timesharing System, ArpaNet, EMACS ("Editor MACroS")
- ▶ 1981 - spin-off Symbolics přebíral většinu původních hackerů a ti podepsali smlouvu na pomoc s vývojem Lisp Machine (LMI). Nové věci se nesmí zpět do MIT verze kopírovat ale dohoda o drobnostech a vzájemném review vývoje.
- ▶ 1982 - MIT přešlo na uzavřený systém nekompatibilní s RMS
- ▶ Později VAX nebo 68020, NDA a slib nepomůžeš bližnímu svému. Když pomůžeš jsi pirát.
- ▶ Nemohu si opravit tiskárnu, jediné NDA a pak opustit SW vývoj
- ▶ 1982-1983 - „Symbolics War“, špehování, právníci zákazy, Stallman pokračuje na MIT Lisp Machine stejně rychle jako 14 vývojářů zavřené verze

- ▶ Jsem přece systémový programátor, napíšem si systém nový GNU's not Unix!
- ▶ 1984 – RMS opouští MIT aby jeho práce na GNU systému nemohla být nabízena MIT pod nesvobodnou licencí
- ▶ 1985 - založena Free Software Foundation
- ▶ Převzaté TeX a X Window systém s dostatečně svobodnou licencí
- ▶ Vlastní EMACS, GCC, GDB, většina potřebných knihoven a Unixových utilit



Svobodný software je software, který respektuje svobodu svých uživatelů a poskytuje jim čtyři základní svobody, které svobodný software definují (publikace FSF 1986):

0. svoboda používat program za jakýmkoliv účelem
1. svoboda zkoumat a upravovat program (předpokladem je přístup ke zdrojovému kódu)
2. svoboda šířit původní verzi programu
3. svoboda šířit upravenou verzi programu

Do 70-tých let běžné, v 80-tých letech nástup copyrightu a restriktivních licencí, povinná četba MS EULA (End-user license agreement) pro všechny, kdo je používají.

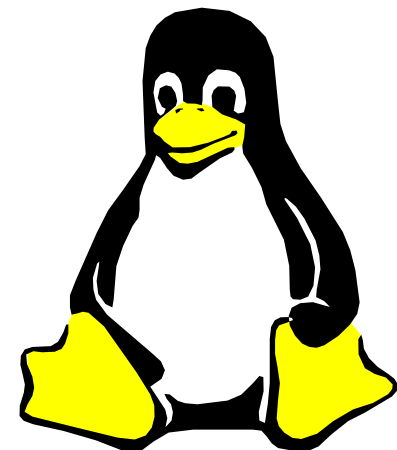
Také rozdíl mezi *free-libre (freedom)* a *free-gratis (zero price)*

Další zdroj např. M. Dočekal http://www.poznejlinux.cz/terminy/svobodny_software

- ▶ 5.1. 1991 si finský student Linus Torvalds kupuje IBM PC s procesorem Intel 80386
- ▶ Po měsíci ho hra Prince of Persia přestane bavit, zkusí Minix, zkusí si napsat přepínání dvou vláken (AAAABBBBAAAA)
- ▶ Emulátor terminálu na čtení pošty
- ▶ V srpnu 1991 Linus informuje o záměru napsat jádro systému na síti Usenet (just a hobby, wont be big and professional like GNU)
- ▶ V září 1991 je verze 0.01 publikována na internetu
- ▶ Složeno s již dokonale připraveným GNU prostředím, vzniká GNU/Linux
- ▶ V červnu 1993 je k dispozici první kompletní distribuce Slackware
- ▶ V březnu 1994 je vydané jádro Linux verze 1.0

- ▶ 2002 – Linus se dohodne s Larry McVoy, že zkusí používat na vývoj jádra BitKeeper. Vzniká společnost BitMover Inc. Linus souhlasí, že je správné použít nejlepší nástroj a nevadí, že je komerční a licence nesvobodná. RMS varuje.
- ▶ BitKeeper k použití zdarma, ale licence se postupně utahuje a zakazuje nejen reverzní inženýrství, ale i práci uživatelů na vývoji jakéhokoliv VCS
- ▶ Duben 2005, BitMover se na základě snahy jiných o plný přístup k datům v historii projektu Linuxového jádra rozhodne odepřít volný přístup k programu. Do noty jim hraje chyba v čítání verzí a blížící se commit 65536.
- ▶ Linus na dva měsíce zastavuje vývoj jádra a ze skriptů v Bashi a pár kousků Céčka si skládá GIT.
- ▶ 17.4.2005 12:20:36 AM – commit Linux v2.6.12-rc3 a jedeme dál

- ▶ Linus Torvalds: Zobrazit náhled výšivky v PES formátu
- ▶ Řešení:
 - ▶ Formát je sice nedokumentovaný, ale již OSS v C# a jiný v PHP
 - ▶ Tak interpretaci přepíše do C (je mi asi nejbližší) a snadno se propojí s grafickou knihovnou Cairo (<http://www.cairographics.org>) a ta již vykreslí výstup do PNG
- ▶ <http://torvalds-family.blogspot.com/2010/01/embroidery-gaah.html>
- ▶ <http://git.kernel.org/?p=linux/kernel/git/torvalds/pesconvert.git;a=summary>
- ▶ [git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/pesconvert.git](http://git.kernel.org/pub/scm/linux/kernel/git/torvalds/pesconvert.git)



Příspěvek Jim Zemlin (Linux Foundation Executive Director) na konferenci TEDx organizace TED (Technology, Entertainment and Design)

▶ Don't Dream Big

- ▶ “Don't aim for success if that's what you want. Do what you love and believe in and it will follow.”

▶ Give It All Away

- ▶ Linus Torvalds a komunita kolem OS Linux vytvořila nesmírné hodnoty (odhad \$10B). Akcie IBM a Red Hat-u stále stoupají na rozdíl od firem za uzavřenými systémy. I Apple a další znají cenu open-source (WebKit, GCC a nyní LLVM)

▶ Don't Have a Plan

- ▶ Zájem inovovat, potřeby aplikací, uživatelé a komunita jsou nejlepším motorem inovací a jsou schopní se organizovat sami.

▶ Don't Be Nice

- ▶ Diskuze, kritika, argumentace bez servítek a nutnost argumenty obhájit své řešení ⇒ lepší řešení než přílišná ohleduplnost nebo i brainstorming.

Talk is cheap. Show me the code.