Portál pre odborné publikovanie ISSN 1338-0087

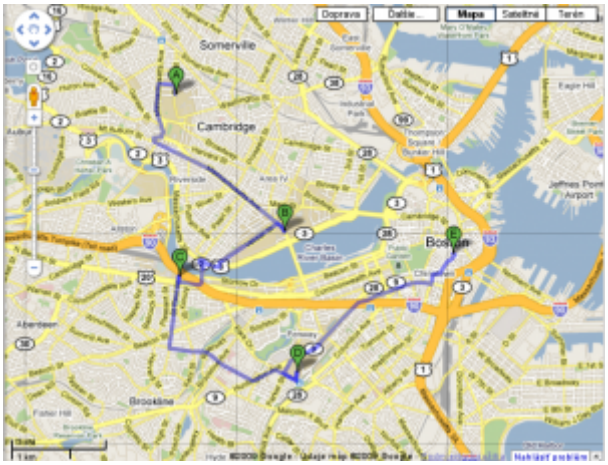
História a princíp UNIX-u 2. časť

Fodrek Peter · Informačné technológie

29.01.2010

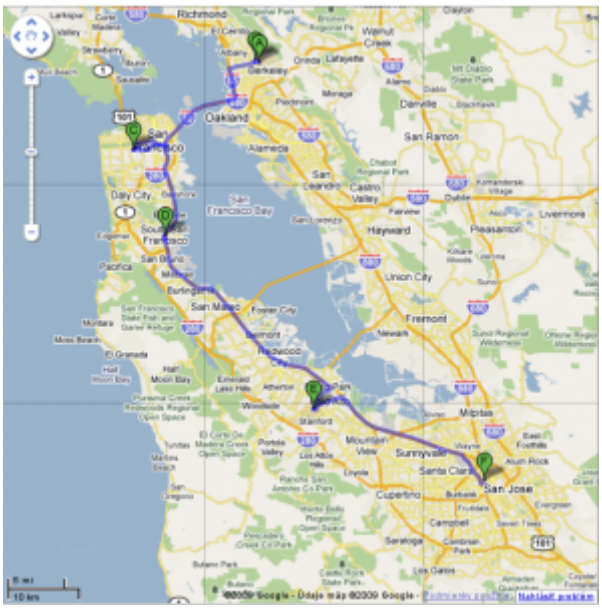
Ako sme si povedali v minulej časti, Unix vznikol v septembri 1969. Keďže sa rozvinul v 70. rokoch a čas v Unixe sa počíta od polnoci z 31.12.1969 na 1.1.1970 v greenwichskom čase, je tento čas označovaný aj ako čas „Doby Unixovej“. Unix bol vďaka jazyku C v roku 1972 portovaný na rôzne hardvérové platformy. Bolo to prvýkrát v histórii, čo bolo možné použiť jeden operačný systém na viacerých typoch počítačov. V tom istom roku sa zrušil greenwichský čas a nahradil sa univerzálnym časom, ktorý z neho vychádza. Aká symbolika. Aby toho nebolo dosť, jedným z posledných detí „predunixovej doby“ bol Linus Benedikt Torvalds, ktorý sa narodil 28.12.1969, a ktorý vstúpil na scénu až o 22 rokov neskôr.

Potenciál Unixu v AT&T neodhadli, preto umožnili prístup k zdrojovým kódom univerzitám. Najviac sa na vývoji podieľali tri univerzity: Massachusets Institute of Technology, University of California at Berkeley a Univerzita Leelanda Stanforda mladšieho. MIT sídli v „Petržalke Bostonu“ meste Cambridge, MA. Táto 105 tisícová obec má až 14 univerzít. Jedenásť je kategórie college, čo zodpovedá našej odbornej vysokej škole (môže udeľovať len bakalárske tituly). Dve z troch zvyšných univerzít sú výskumné univerzity. Na druhej strane Karlovej rieky, v Bostone sídli Free Software Foundation. FSF vytvorili ľudia, ktorí na MIT urobili GNU. GNU bol projekt na vytvorenie operačného systému, ktorý bol nebol Unixom, ale bol ako Unix. Preto je GNU rekurzívna skratka z „GNU is not UNIX“. Paradoxne GNU je dnes jedným zo štandardov Unix-ových systémov a dal svetu GNU Compiler Collection, známe to, **gcc**. Problémom GNU je dodnes neexistencia dokončeného jadra HURD, a tak GNU musela zachrániť Európa. GNU bol naštartovaný v roku 1984.



Obr. 1. Boston a Cambridge,MA: a) Harvardova Univerzita, b) MIT, c) Bostonská univerzita, d) Severovýchodná Univerzita, e) FSF

Na opačnom konci USA, v severnej Kalifornii, sa nachádza druhá centrála vývoja Unix-u. Ide o miesto historicky známe ako severná polovica údolia Svätej Kláry (Santa Clara Valley) a opačný breh Sanfranciského zálivu. Dnes je severná polovica údolia Svätej Kláry známejšia ako Kremíkové údolie (Silicon Valley). Je to oblasť medzi južným San Franciskom a San José (body D a F na obr. 2.)



Obr. 2 Silicon Valley a jeho univerzity: A) UC @ Berkeley, B) Carnegie Mellon Silicon Valley, C) UC @ San Francisco, D) Južné San Francisco= severná hranica SV. E) Stanford, F) San José= južná hranica SV

Geografickým centrom Sanata Clara Valley je Sunnyvale, správnym centrom je Palo Alto, kde sídli IT fakulta univerzity Leelanda Stanforda mladšieho. Nápad dať sem technologické firmy skrsol u miestnych ľudí. Tí boli nespokojní so skutočnosťou, že absolventi Stanfordu chodia robiť do údolia rieky Hudson v štáte New York. Tam totiž sídlia zvyšné technologické firmy ako IBM. Tie dnes mozgami zásobujú len bostonské univerzity. Tak to bolo v 30. rokoch 20. storočia. Názov Silicon Valley sa zrodil v hlave miestneho vizionára a prvý raz bol zverejnený v novinách dňa 11.1.1971.

Skutočnosť, že Unix bol vyvíjaný na viacerých špičkových univerzitách spôsobilo, že nie je a nikdy nebol jednoduchý. Potvrdil to aj Linus Torvalds vo svojeje odpovedi na otázku: *"Nezdá sa Vám, že Linux opúšťa Unixové tradície a začína byť zbytočne komplikovaný ?"*. Odpovedal : *"Vlastne, kúzlo Unixu nieje v tom, že by vôbec niekedy bol jednoduchý. Je to o tom, že potrebujete mať nejaké koncepty na vysokej úrovni. Tie sú dôležité a určujú dizajn. Koncepty ako 'jednotný priestor pre súbory, všetky dáta sú tok bytov a všetko je proces, sú stále pravdivé napriek tomu, že čas zmenil veľa detailov. Takže detaily sú komplikované."*

Nástroje UNIX-u koncepčne vychádzajú z metódy dekompozície. Jednoducho máme veľa nástrojov, ktoré robia jednu veľmi konkrétnu vec v mnohých modifikáciách a vždy presne. Preto má každý príkaz UNIX-u *„miliónpäť"* prepínačov, alebo

kombinácií prepínačov. Na riešenie zložitejších problémov však treba mať systém komunikácie medzi príkazmi a programami. Toto má Unix vyriešené ukážkovo a preto ho akademici *„zbožňujú“*. Bežným používateľom však niektoré pretechnizované a príliš akademické koncepty nie sú blízke.

Prvý textový editor v UNIX-e bol **ed**. Ten bol a stále je v UNIX-e od roku 1969. Bol to riadkový editor. Napísal ho Ken Thomson a bol prvým editorom na svete s regulárnymi výrazmi. Bol inšpirovaný editorom **QED** z Berkeley, kde Ken študoval. Vo vývoji Unixu sa začínajú objavovať dva hlavné prúdy. Jeden vychádzal zo základov AT&T a značil sa názvom System s rímskym číslom (System I, System II, System III, System IV a dnes System V). Verzie Kalifornskej univerzity v Berkeley zas ako BSD (Berkeley Software Distribution). V BSD sa dočkal **ed** nástupcu v editore **ex** z EXtended. Kópia editora **Ex** v MS-DOS-e sa volal **EDLIN**, čo možno viac zdôrazňuje, že ide o riadkový editor. Ex bol vydaný ako súčasť BSD v roku 1977. Už v roku 1976 vznikol jeho nástupca editor **vi** (Visual). Ten využíval systém **termcap**, ktorý sa dostal do System III ako **terminfo**. Bol, a je, to systém na písanie aplikácií, ktoré bežia na rôznych termináloch na celej obrazovke a vyzerajú rovnako aj keď terminály majú rozdielny protokol komunikácie a rozdielne možnosti (farby, veľkosť písmen, font, obnovovacia frekvencia monitora…). Tieto informácie si zisťoval a udržiaval systém sám. Aplikácia využívajúca **termcap/terminfo** vedela údaje čítať a poľa nich sa správať. To bolo nepohodlné a navyše **termcap** nepodporoval farby a ťažko sa teda písali programy univerzálne pre všetky Unix-y. Toto vyriešila nadstavbová knižnica **curses**, ktorá poskytovala jednotné API (Application Programing Interface) / ABI (Application Binary Interface) na tvorbu programov. Program sa napísal raz a bežal všade a to bez nutnosti, aby autor programu poznal všetky terminály. Rozdiely medzi **termcap** a **terminfo** ako aj Unixami všeobecne donútili vývojárov zaviesť dynamicky linkovateľné a zdieľané knižnice (vo Windows známe ako DLL), ktoré sa v Unix-och volajú shared objects (súbory s menom obsahujúcim .so). Vďaka tomu sa editor **vi** dostal do BSD2 v roku 1978. Keď bol vývoj pôvodnej knižnice **curses** v polovici 90. rokov 20. storočia zastavený a bol pokus nechať len grafickú knižnicu **X/Open Curses**, tak sa našla skupina, ktorá začala vyvíjať knižnicu **ncurses** (new Curses), ktorá je plne kompatibilná s pôvodnou. Vďaka tomu je editor **vi**, alebo jeho verzia **vim** (VIsual IMproved) dostupná na každom modernom UNIX-e. Preto je dobré si niečo povedať o tomto editore.

Základom sú dva režimy práce. Príkazový a editačný. V príkazovom režime sa logicky zadávajú príkazy a v editačnom sa mení text. Z editačného do príkazového režimu sa dostaneme stlačením klávesy resp. . Klávesa Escape nemá v príkazovom režime žiadnu funkciu. V príkazovom režime zadávame príkazy editora vi/vim, ktoré sú jedno- a dvoj-znakové. Ak však ako prvý znak zadáme „:“ zadávame za ním postupnosť jedno-písmenové príkazy editor ex. Tie sa vykonajú až keď sa postupnosť ukončí stlačením klávesy . Najdôležitejšie príkazy sú :

- w zapíše súbor na disk
- q ukončí program ak je súbor uložený alebo nebol zmenený, inak varuje, že súbor nebol uložený a neurobí nič
- ! ignoruje varovania predchádzajúceho príkazu a zabezpečí jeho vykonanie napriek možným následkom

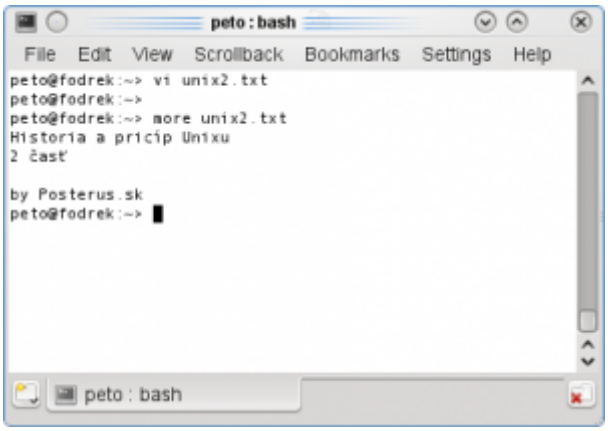
Ak teda chceme ukončiť program bez uloženia zmien použijeme kombináciu :q!. Do editačného režimu sa dosteneme z príkazového rôznymi príkazmi editora vi. Najčastejšie príkazy na prechod do editačného režimu sú :

- i vloží text na miesto kde je kurzor
- dd – zmaže aktuálny riadok
- dw – zmaže aktuálne slovo

Pre ilustráciu si prácu s vi editorom ukážeme na jednoduchých príkladoch. V prvom kroku si vytvoríme nový súbor unix2.txt (prvý príkaz na obr. 4). Stlačením **:i** môžeme písať text do súboru (obr. 3)

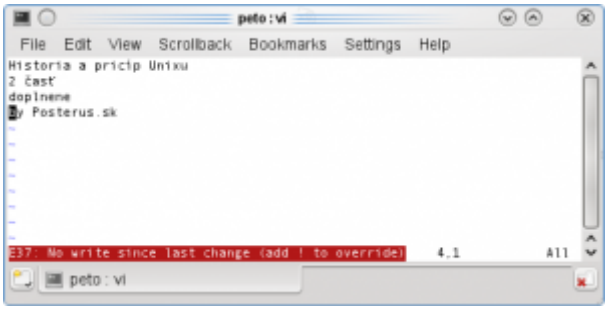
Obr. 3 Editačný režím editora vi

Keď napíšeme náš text, do príkazového režimu sa dostaneme klávesou Escape. Súbor uložíme pomocou príkazov **:wq!**. Súbor si môžeme následne vypísať príkazom **more** (obr. 4).



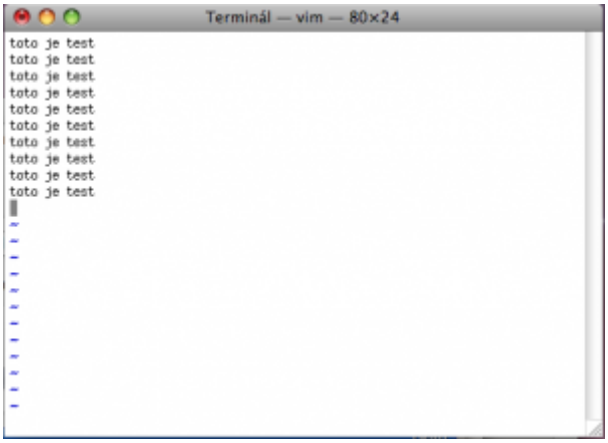
Obr. 4 Vytvorenie súboru a jeho výpis

Ak by sme chceli dopísať ďalší text, znovu si otvoríme súbor *unix2.txt* a pomocou príkazu :i sa dostaneme do editačného režimu. Stalačením klávesy Escape sa dostaneme do príkazového režimu. Teraz vyskúšame opustiť editor bez ignorovania upozornení pomocou príkazu **:q**. Ako je vidieť na obrázku 5, editor nás upozorní pomocou chybovej správy.



Obr. 5 Ukončenie editora vi s chybou

Pred príkaz možno dať číslo, ktoré značí počet opakovaní daného príkazu, teda napr. 3dd zmaže 3 riadky od kurzora smerom dole. Alebo 10i toto je test napíše 10x Toto je test, pričom každý výpis je na novom riadku (obr. 6). Pozor, Unix rozlišuje veľké a malé písmená. Preto sú I a i vo vi editore rôzne príkazy.



Obr. 6 Výsledok vo vim v grafickom terminále

Pri editoroch v textovom režime, ktorých je omnoho viac (joe, pico/nano, atď.) platí, že ak napíšeme za meno editora názov súboru, tak sa otvorí ten súbor v editore. Pri editoroch sme sa zmienili aj o **(n)cruses** a **termcap**. Oba sú aj v MS-DOS-e. Keďže MS-DOS má len konzolu a aj to len jediného typu, používa sa zjednodušená verzia **curses** s názvom **CONIO** (CONsole Input/Output). Prečo je však bod so súradnicami [0,0] interpretovaný ako ľavý horný a nie ako by logika kázala ľavý dolný bod ? A odkiaľ píše písací stroj, teda aj ďalekopis? A sme doma. Dôvodom je ľahká interpretácia na termináloch. Preto rovnakým spôsobom chodí aj lúč na CRT monitoroch a televízoroch. Prečo sa ten lúč vracia vždy na začiatok riadku a nekreslí každý druhý riadok od konca? Unix na prechod medzi riadkami používal ASCII znaky LF (Line feed =posun na nový riadok) a CR (Carriage return =návrat vozíka). Ak ste videli písací stroj, tak viete, že na nový riadok sa posúval pákou. To je vykonanie LF. Potom bolo treba valec posunúť doprava to je CR. Unix, kvôli nedostatku pamäte v 60. rokoch ukladá v textových súboroch len LF a CR pridáva pri výpise. Na to pridanie však vtedy potreboval čas a ten mu dal návrat lúča. Niektoré editory v MS Windows (Notepad) nepridávajú CR a preto v nich súbory robené na Unixe vyzerajú nepekne (obvykle sú nesprávne zalomené). Ďalšia vec, ktorú Microsoft nemá dodnes, je možnosť mať niekoľko rôznych dynamických knižníc s rovnakým menom v rovnakom adresári a systém na vybratie tej správnej. Úplne iná vec je príkaz **sudo**, ten príkaz umožňuje prepnúť sa na vykonanie jedného príkazu na iného užívateľa. Do Unix-u sa dostal v roku 1980. Do MS Windows sa pod menom „**runas**“ dostal v NT5.0 ktoré sa predávali ako MS Windows 2000. Tu môžeme postrehnúť odlišnú koncepciu

oboch operačných systémov. Zatiaľčo Unix od začiatku bol navrhovaný ako viacpoužívateľský, tak Windows si zakladal práve na tom, že je inštalovaný na osobné počítače. Každý používateľ teda pracoval lokálne na svojom systéme. Potreba rozlíšenia práv používateľa bola minimálna. Grafické nadstavby **sudo** boli v Unixe už pred uvedením **KDE** (rok 1998). Do MS Windows sa dostali vo Windows VISTA ako User Account Control (UAC), ale užívatelia ho odmietli a snažili sa ho vypnúť. Pritom ide o zásadný posun v bezpečnosti. Tým sme skočili rozprávanie o textových programoch. K nim ešte patrí, ale pár detailov. Vďaka **curses** vznikli súborové manažéry Midnight Commander v Unixe a Norton Commander a Total Comander na OS Microsoftu.

Jedna vec, ktorú Unix vysvetľuje je nutnosť pri volaní funkcie jazyka C **exec***, zadávať medzi cestu ku programu a parametre aj takzvaný nultý parameter, ktorý je zhodný s menom programu. Volanie **exec** spustí program zo súboru na disku a tak sa zdá parameter ako duplicitný. V System II a novších je príkaz na výpis dlhých súborov po stránkach/obrazovkách „**more**“, v BSD to robí príkaz „**less**“. V GNU vyrobili jeden program, ktorý raz nahrali na disk a ktorý plnil funkcie ako more tak less, keďže sa mierne líšili parametrami. Vytvorili link na súbor toho programu nahraný ako more, ktorý mal meno less, a podľa toho, ktorým príkazom bol program volaný menil spracovanie parameterov. To, ktorým príkazom bol program volaný sa prejavilo práve zmenou obsahu nultého parametra. V Unix-e existuj[dva typy linkov – symbolické a tvrdé. Symbolické linky prebral Microsoft ako zástupcov. Tvrdé linky zabezpečujú, aby sa súbor nezmazal, pokiaľ na neho existuje aspoň jeden tvrdý link. Pri hard linkoch teda neexistujú neplatné odkazy známe z Microsoft Windows. Bohužiaľ hard linky Microsoft dodnes nevie vytvárať.

Ďalšou zaujímavosťou knižnice **curses**, je že sa výstup zobrazí až po volaní funkcií refresh/wrefresh. Je to preto, lebo už vtedy sa šetrilo papierom (mohlo by sa začať opäť) a ak by nebol refresh, tlačil by sa pri zmene každého jediného znaku celý list papiera. Refresh teda znamená dokončil som zmenu obrazovky, môže sa to vytlačiť.

Keďže grafické programy v Unix-e sú prevažne programy, ktoré uľahčujú prácu s ich textovými obdobami povieme si nabudúce o tom ako funguje grafika v Unixoch a začneme s ním pracovať.