Sciabarrà .com

Symbian Java Linux

Chi ha scritto davvero Linux?

La notizia di questi giorni è che un certo Ken Brown, presidente di una fondazione nota come **Alexis de Tocqueville Institution**, sostiene in uno studio (peraltro accessibile solo a pagamento) tra le altre cose, che Linux non è stato scritto da Linus Torvalds, ma invece è stato plagiato da Minix, il sistema operativo a cui si è dichiaratamente ispirato. La notizia è infamante e, per chi conosce un minimo di storia di Linux, palesemente falsa. Ma una smentita chiara e particolareggiata è necessaria anche per il grande pubblico.

La persona più indicata per smentire questa affermazione è proprio l'autore del sistema "plagiato", Minix, ovvero Andrew Tanenbaum. Il "maestro" di Linus (e anche mio) nel suo sito ha pubblicato una interessante memoriale sull'argomento, evidenziando l'ignoranza e i preconcetti sull'argomento di Ken Brown, e l'opacità delle sue motivazioni (che non avrebbe motivo di nascondere, SE non fossero sospette).

Ho contattato per email proprio Andrew Tanenbaum chiedendo ed ottenendo l'autorizzazione a tradurre in italiano le sue pagine al riguardo. Questo allo scopo di fornire al lettore sia una chiara smentita di argomentazioni false, sia una visione storica e interessante delle origini di Linux. Per esempio, del fatto che sia effettivamente sia possibile, per una sola persona, scrivere un **piccolo** sistema operativo (e Tanenbaum cita almeno 6 persone che hanno fatto una cosa simile). Notare che queste dichiarazioni vengono proprio da una persona che ha effettivamente scritto un sistema operativo (anzi due), per di più dichiaratamente lavorando da solo la sera.

La storia raccontata da Tanenbaum è anche un eccellente e affascinante complemento al testo storico che ho già scritto sulla storia di Linux e che **può essere scaricata da qui**.

I fatti narrati mi coinvolgono personalmente. Ho letto l'intero codice sorgente di Minix durante un mio esame universitario, e ne ho tratto una passione immensa per Unix prima e Linux dopo. Ho installato Minix sul mio 8086 anni fa e ho sognato di poter disporre di un sistema completo, trasferendomi praticamente in pianta stabile al centro di calcolo dell'università per poter usare un VAX con Ultrix.

Sono onoratissimo dell'autorizzazione ottenuto di poter tradurre un testo del maestro, Andrew Tanenbaum, e confermo come l'impresa di **avvio** di Linux sia realmente fattibile per un "hacker dedicato" che abbia le necessarie competenze e capacità. Il resto è storia, ed è il risultato di una vasta comunità che si è creata intorno a un progetto realmente valido.

Banner incluso su richiesta di Andy Tanenbaum

Alcune note sulla *Kerfuffle* "Chi ha scritto Linux", Release 1.5

Scarica Omaggio
il Capitolo 2

Leggi Online
il Capitolo 6

(**Kerfuffle**: confusione, disturbo)

di Andrew Tanenbaum **tradotto in italiano dall'originale** da Michele Sciabarrà

Antefatto

La storia di UNIX e dei suoi vari figli e nipoti è stata recentemente oggetto di attenzione come conseguenza di un libro dalla Alexis de Tocqueville Institution. Poiché sono stato coinvolto in una parte della storia, mi sento obbligato a fornire alcuni chiarimenti e correggere degli errori molto seri. Ma prima forniamo alcune informazioni sull'antefatto.

Ken Brown, presidente della Alexis de Tocqueville Institution, mi ha contattato all'inizio di Marzo. Mi ha detto che stava scrivendo un libro sulla storia di UNIX e gli sarebbe piaciuto intervistarmi. Poiché ho scritto 15 libri e sono stato coinvolto nella storia di UNIX in molti modi, gli ho risposto che sarei stato lieto di aiutarlo. Sono stato intervistato da molte persone per varie ragioni nel corso degli anni, e sono comparso nelle televisioni tedesca e statunitense, e su vari giornali e riviste; pertanto non ci ho pensato molto sopra, al riguardo.

Brown è arrivato in aereo ad Amsterdam per intervistarmi il 23 marzo 2004. Apparentemente, la mia intervista era la sola ragione per la sua venuta in Europa. L'intervista ha avuto un inizio brusco, all'incirca parafrasato come segue:

AST: "Che cosa è la Alexis de Tocqueville Institution?"
KB: Facciamo del lavoro politico pubblico.
AST: Una specie di "think tank", come la Rand Corporation?
KB: Qualcosa del genere.
AST: Che cosa fa?
KB: Pubblica rapporti e libri.
AST: Chi la finanzia?
KB: Abbiamo molteplici fonti di finanziamento.
AST: SCO è una di quelle? E' correlato alla causa (contro Linux ndt) di SCO?
KB: Abbiamo molteplici fonti di finanziamento.
AST: Microsoft è una di quelle?
KB: Abbiamo molteplici fonti di finanziamento.

Egli fu estremamente evasivo a proposito del perché era lì e chi lo stava finanziando. Continuò a dire che stava solamente scrivendo un libro sulla storia di UNIX. Gli chiesi cosa pensava del libro di Peter Salus, "Un quarto di secolo di UNIX". Non ne aveva mai sentito parlare! Se tu stai scrivendo un libro sulla storia di UNIX e stai facendo un volo di 3000 miglia per intervistare una persona su un argomento, non avrebbe senso almeno andare su amazon.com, digitare "history unix" nella ricerca, ottenendo in quel caso il libro di Salus come primo risultato? Per $28 (e invio gratuito, se giochi bene le tue carte) puoi imparare un sacco di cose sul materiale oggetto del tuo studio, e non avere nessuno sfasamento da fuso orario. Come ho capito presto, Brown non è il coltello più affilato del cassetto, ma io ero già sospettoso. Poiché sono un autore da lungo tempo, so che ha senso almeno conoscere la concorrenza. A lui non gliene importava.

Io e UNIX

Non mi è sembrato strano che Brown volesse intervistarmi a proposito della storia di UNIX. Ci sono persone peggiori a cui chiedere. Nei tardi anni '70, e nei primi anni '80, sono stato parecchie estati nel gruppo UNIX (Dipartimento 1127) ai Laboratori Bell. Ho conosciuto Ken Thompson,

Dennis Ritchie, e il resto delle persone coinvolte nello sviluppo di UNIX. Sono stato a casa di Rob Pike e Al Aho per lunghi periodi di tempo. Dennis Ritchie, Steve Johnson e Peter Weiberger, tra gli altri, sono stati nelle mia casa di Amsterdam. Tre dei miei studenti di dottorato hanno lavorato nel gruppo UNIX ai Laboratori Bell, e uno di loro è ora un membro permanente dello staff.

Stranamente, quando sono andato ai Laboratori Bell, il mio interesse non erano i sistemi operativi, anche se io ne avevo scritto uno e pubblicato un articolo a proposito (vedi "Software - Practice & Experience," vol. 2, pp. 109-119, 1973). Il mio interesse erano i compilatori, poiché ero il principale progettista dell'Amsterdam Compiler Kit (vedi Commun. of the ACM, vol. 26, pp. 654-660, Sept. 1983). Ho trascorso del tempo presso di loro, discutendo di compilatori con Steve Johnson, di reti con Greg Chesson, sviluppando strumenti con Lorinda Cherry, e scrivendo libri con Brian Kernighan, tra l'altro. Divenni anche amico di diversi "stranieri" in visita in quel luogo, tra cui Bjarne Stroustrup, che più tardi sarebbe diventato il progettista e iniziale implementatore del C++.

In breve, sebbene io non avessi avuto niente a che fare con lo sviluppo dello UNIX originale, ho conosciuto tutte le persone coinvolte, e so abbastanza bene molto della storia . Per di più, i miei contatti con il gruppo UNIX ai Laboratori Bell non erano un segreto; li ho perfino ringraziati pubblicamente per avermi avuto come visitatore estivo nella prefazione alla prima edizione del mio libro Computer Networks. La cosa stupefacente era che Brown non sapeva niente di tutto questo. Non aveva fatto i compiti a casa prima di imbarcarsi nel suo piccolo progetto.

Io e MINIX

Anni dopo, stavo insegnando un corso sui sistemi operativi e stavo usando come testo il libro di John Lions su UNIX versione 6. Quando la AT&T; decise di proibire l'insegnamento del funzionamento interno di UNIX, ho deciso di scrivere la mia versione di UNIX, libera di tutto il codice e di tutte le restrizioni imposte dalla AT&T.; La mia ispirazione a scrivere una nuova versione di UNIX non furono le visite ai Laboratori Bell, poiché ero a conoscenza del fatto che una persona poteva scrivere un sistema operativo simile a UNIX (Ken Thompsono scrisse UNICS su un PDP-7), sapevo che si poteva fare. La mia reale ispirazione fu una nota di Butler Lampson in un corso sui sistemi operativi che avevo seguito quando ero uno studente di dottorato a Berkeley. Lampson aveva appena finito di descrivere le origini del sistema operativo CTSS e disse, nel suo modo inimitabile: "C'è qualcuno di voi che non saprebbe scrivere CTSS in un mese?". Nessuno ha alzato la mano. Ho concluso che sarei stato realmente stupido a non saper scrivere un sistema operativo in un mese. Il progetto citato prima è un sistema operativo che ho scritto a Berkley con l'aiuto di Bill Benson. Ci è voluto molto più di un mese, ma io non sono così bravo come Butler. Nessuno lo è.

Decisi di scrivere un clone minimale di UNIX, MINIX, e lo feci da solo. Il codice realizzato era libero al 100% dalla proprietà intellettuale della AT&T.; L'intero codice sorgente è stato pubblicato nel 1987 nell'appendice di un libro, "Sistemi Operativi: Progettazione e Implementazione", che più tardi ebbe una seconda edizione, con Al Woodhull come co-autore. MINIX 2.0 era perfino conforme a POSIX. Entrambe le edizioni contenevano centinaia di pagine di testo che descrivevano il codice con dovizia di particolari. Una scatola di 10 floppy disk conteneva tutti i binari, e il codice sorgente era disponibile separatamente dalla Prentice Hall per $69.

Mentre non si trattava di "free software" nel senso di "birra gratis" (ovvero software gratis, ndt), era "free software" nel senso di "libertà di parola" (ovvero software libero, ndt), poiché tutto il codice sorgente era disponibile per un costo leggermente superiore a quello delle spese di gestione. Ma perfino la libertà di parola non è completamente libera --

pensate alla diffamazione, a urlare "al fuoco" in un teatro affollato, eccetera. Inoltre ricordate (se siete abbastanza vecchi) che nel 1987, una licenza per fini formativi di UNIX costava $300, una licenza commerciale per una università $28.000, e una licenza commerciale per una azienda costava molto di più. Per la prima volta, MINIX faceva scendere il costo del codice sorgente di un "simil-UNIX" alla portata delle tasche di uno studente.

Prentice Hall non era realmente interessata a vendere software. Loro erano interessati a vendere libri, per decisero una politica abbastanza liberale sulla possibilità di copiare MINIX; ma se una azienda voleva venderlo per fare molti soldi, PH voleva una percentuale. Per cui gli avvocati di PH aggiunsero a MINIX una gran quantità di zavorra legale, ma non c'era nessuna intenzione di applicarla contro università o studenti. Usare Internet per la distribuzione di tutto quel codice non era fattibile nel 1987: perfino per persone con un modem ad "alta velocità" (per esempio 1200 bps). Quando la distribuzione via Internet divenne possibile, ho convinto Prentice Hall a rinunciare alle sue (estremamente modeste) ambizioni commerciali, e darmi il permesso di pubblicare il sorgente nel mio sito web per il libero download, dove è ancora oggi.

Dopo un paio di mesi dal suo rilascio, MINIX divenne oggetto di culto, con un suo newsgroup USENET, comp.os.minix, con 40.000 iscritti. Molte persone aggiunsero nuovi programmi di utilità e migliorarono il kernel in molti modi, ma il kernel originale era il lavoro di una sola persona -- io. Molte persone iniziarono a incalzarmi chiedendomi di migliorarlo. In aggiunta ai numerosi messaggi nel newsgroup di USENET, mi arrivavano 200 email al giorno (ai tempi in cui solo pochi scelti avevano del tutto un e-mail), dicendo cose come "mi servono i pseudoterminali, e mi servono per venerdì". La mia risposta era generalmente veloce e andava al punto: "No.".

La ragione del mio frequente "no" era legata al fatto che tutti stavano tentando di trasformare MINIX in un sistema UNIX di qualità adeguata per sistemi di produzione; io non volevo che diventasse qualcosa di così complicato per il mio obiettivo, ovvero insegnarlo agli studenti. Inoltre mi aspettavo che la nicchia per un sistema UNIX "free" (libero e/o gratuito ndt) di qualità adatta per la produzione sarebbe stato riempito da GNU o dallo UNIX Berkeley in breve tempo; per cui non stavo realmente mirando a questo. Come risultò poco dopo, l'OS GNU non andò da nessuna parte (sebbene molte utility erano state già scritte) e lo UNIX di Berkeley rimase impigliato in una causa legale; questo fu quando i suoi progettisti fondarono una ditta, la BSDI, per venderlo e scelsero 1-800-ITS-UNIX come numero di telefono (It's Unix: è Unix ndt). AT&T; concluse che questo costituiva una violazione di copyright, e gli fece causa. Ci vollero un paio di anni perché questa disputa venisse risolta. Questo ritardo nell'avere un BSD libero diede a Linux il respiro necessario che gli servì per raggiungere il suo successo planetario. Se non fosse stato per la causa legale, indubbiamente BSD averebbe riempito la nicchia per un clone di UNIX potente e libero, poiché era un sistema già stabile, maturo e con un largo seguito.

Io e Ken Brown

Quando Ken Brown svelò le sue intenzioni e iniziò a fare domande, io avevo rapidamente capito che non conosceva niente della storia di UNIX, non aveva mai sentito parlare del libro di Salus, e non sapeva nulla di BSD e della causa della AT&T.; Iniziai a raccontargli la storia, ma mi fermò e mi disse che era più interessato agli aspetti legali. Gli dissi: "Oh, intende parlare del brevetto di Dennis Ritchie numero 4135240 sul bit setuid?" E aggiusi "Non è un problema. I Laboratori Bell hanno dedicato il brevetto". A quel punto scoprii che (1) non aveva mai sentito parlare di brevetti (2) non conosceva che cosa significava dedicare un brevetto (ovvero renderlo di pubblico dominio), e (3) non sapeva davvero nulla sulle leggi relative alla proprietà intellettuale. Era confuso a proposito di brevetti, copyright e

trademark. Gli chiesi se era un avvocato, ma era ovvio che non lo era e lo ammise. A quel punto stavo ancora pensando che potesse essere una spia da SCO, ma se lo era, SCO non stava spendendo bene i suoi soldi.

Egli voleva parlare di questioni relative alla proprietà intellettuale, ma stava anche cercando di evitare quale reale scopo avesse, per cui non formulava le sue domande molto bene. Finalmente mi chiese se io pensassi che Linus avesse scritto Linux. Gli dissi che al meglio della mia conoscenza, Linus aveva scritto l'intero kernel da solo, ma dopo che era stato rilasciato, altre persone avevano iniziato a migliorare il kernel, che era inizialmente molto primitivo, e aggiungere nuovo software al sistema -- essenzialmente lo stesso modello di sviluppo di MINIX.

A quel punto iniziò a mettere a fuoco su questo argomento, con domande quali "non ha rubato pezzi di MINIX senza permesso?". Gli ho detto che MINIX ha avuto chiaramente una enorme influenza su Linux in molti modi, dalla struttura del file system ai nomi nel codice sorgente, ma non pensavo che Linus avesse usato parte del mio codice. Linus inoltre usò MINIX come sua piattaforma iniziale di sviluppo, ma non c'era niente di sbagliato in questo. Egli chiese se io avessi mosso obiezioni sul fare questo, e gli dissi di no, non lo avevo fatto, la gente era libera di usarlo come desiderava per fini non commerciali. Più tardi MINIX fu rilasciato sotto la licenza di Berkeley, che lo rese libero per ogni scopo. E' ancora sorprendentemente in largo uso, sia per scopi formativi che nel Terzo Mondo, dove milioni di persone sono felici come una pasqua di avere un vecchio 386 con 1 MB di RAM, sui quali MINIX gira benissimo. La pagina di MINIX citata prima viene visionata ancora 1000 volte la settimana.

Finalmente, Brown inizio a mettere a fuoco chiaramente. Continuò a chiedere, in varie forme, come una persona possa scrivere un sistema operativo tutto da solo. Semplicemente non credeva che fosse possibile. Per cui dovetti raccontargli altra storia, sigh. Per cominciare, Ken Thompon scrisse UNICS per il PDP-7 tutto da solo. Perfino quando iniziò a lavorare su un PDP-11 e a riscriverlo in C, Dennis Ritchie si aggiunse al team, ma principalmente si concentrò sulla progettazione del linguaggio C, scrivendo il compilatore C, e scrivendo il sistema di I/O e i driver per le periferiche. Ken scrisse quasi tutto il kernel da solo.

Nel 1983 una ditta oggi scomparsa chimata Mark Williams produsse e mise sul mercato un clone di UNIX molto buono chiamato Coherent. Gran parte del lavoro fu fatto da tre ex-studenti dell'università di Waterloo: Dave Conroy, Randall Howard e Johann George. Ci misero due anni. Ma produssero non solo il kernel, ma anche il compilatore C, la shell, e TUTTE le utility UNIX. Questo è molto più del lavoro di fare solamente un kernel. E' probabile che il kernel abbia preso loro meno di un anno-uomo.

Nel 1983 Ric Holt pubblicò un libro, ora non più in stampa, sul sistema TUNIS, uno sistema simile a UNIX. Questo era sicuramente una riscrittura in quanto il TUNIS fu scritto in un nuovo linguaggio, il concurrent Euclid.

Poi Doug Comer scrisse XINU. Sebbene non era un clone di UNIX, era un sistema paragonabile.

Per di più, Gary Killdall scritte il CP/M da solo, e Tim Paterson scrisse MS-DOS. Mentre questi sistemi dei primi anni 80 non erano nemmeno vicini ad essere dei cloni di UNIX, essi erano sistemi operativi concreti e popolari, scritti da singoli individui.

All'epoca in cui Linus aveva iniziato, cinque persone o piccoli gruppi avevano implementato il kernel di UNIX o qualcosa di approssimativo, cioè: Thompson, Coherent, Holt, Comer, e io. Tutto questo era perfettamente legale e nessuno rubò nulla. Data questa storia, è abbastanza difficile fare un caso che una persona non può implementare un sistema della complessità di Linux, la cui originale dimensione era all'incirca la stessa della versione 1.0 di MINIX.

Naturalmente, è sempre vero nella scienza che le persone costruiscono sul lavoro dei predecessori. Perfino Ken Thompsono non fu il primo. Prima di scrivere UNIX, Ken aveva lavorato sul sistema MULTICS del MIT (MULTiplexed Information and Computing Service). Infatti il nome originario di UNIX fu UNICS, un gioco di parole che stava per UNiplexed Information and Computing Service, poiché la versione che girava sul PDP-7 poteva supportare solo un utente -- Ken. Dopo troppe battute sul fatto che EUNUCHS fosse un MULTICS castrato (UNICS si può pronunciare pressappocopo come l'equivalente della parola inglese "eunuco"), il nome fu cambiato in UNIX. Ma perfino MULTICS non fu il primo. Prima fu il summenzionato CTSS, progettato dallo stesso team al MIT.

Quindi, naturalmente, Linus non si sedette in un vuoto e improvvisamente cominciò a scrivere il codice sorgente di Linux. Egli aveva il mio libro, stava usando MINIX, e indubbiamente conosceva la storia (anche perché è nel mio libro). Ma il codice fu il suo. La prova è che ha incasinato l'architettura. (Scusate il termine ma è quello che rende meglio in italiano il termine "mess" e l'intenzione di Tanenbaum, ndt). MINIX era un simpatico e modulare sistema a microkernel, con un gestore della memoria e un file system che venivano eseguiti come processi nello spazio utente. Questo rende il sistema più chiaro e più affidabile di un grosso kernel monolitico, e più facile da correggere e mantenere, con un piccolo prezzo in prestazioni: perfino in un 8088 a 4.77 MHz si avviava in circa 5 secondi (contro un minuto per Windows su un hardware 500 volte più veloce).

Un esempio di un microkernel di successo è QNX. Invece di scrivere un nuovo file system e un nuovo gestore di memoria, che sarebbe stato facile, Linux riscrisse tutto quanto come un grosso kernel monolitico, aggiungendo perfino del codice assembler "in linea" (mescolato al codice C, ndt) :-(. La prima versione di Linux era come una macchina del tempo. Portava indietro a un sistema peggiore di quello che aveva già sulla sua scrivania. Naturalmente era solo un ragazzo e non sapeva far di meglio (ciò nonostante, se avesse fatto maggior attenzione in classe, avrebbe potuto), ma produrre un sistema che era fondamentalmente differente dalla base da cui era partito mi sembra una chiara prova che era una progettazione da zero. Io non penso che abbia copiato UNIX perché non aveva accesso al codice sorgente di UNIX, eccetto forse il libro di John Lions, relativo a una precedente versione di UNIX che non assomiglia molto a Linux.

La mia conclusione è che Ken Brown non sapeva di cosa stava parlando. Io ho anche gravi critiche sulla sua metodologia. Dopo che ha parlato con me, è andato all'università e ha cominciato a fermare studenti a caso e a fare domande. Non esattamente fonti dirette.

Le sei persone che io conosco che hanno (ri)scritto) UNIX hanno fatto tutto in maniera indipendente, e nessuno ha rubato nulla a qualcun altro. La nota di Brown che le persone hanno tentato e fallito per 30 anni di costruire un sistema simile a UNIX è chiaramente senza senso. Sei diverse persone lo hanno fatto indipendentemente uno dall'altro. Nella scienza è considerato importante dare credito alle persone le loro idee, e io credo che Linus ha fatto questo molto meno di quando avrebbe dovuto. Ken e Dennis sono i veri eroi qui. Ma la scarsa tendenza di Linus a dare riconoscimenti non è una ragione per asserire che Linus non ha scritto Linux. Non ha scritto CTSS e non ha scritto MULTICS e non ha scritto UNIX e non ha scritto MINIX, ma ha scritto Linux. Penso che Brown debba a molti di noi delle scuse.

Io e Linus

Alcuni possono trovare strano che io stia difendendo Linus in questa sede. Dopo tutto, io e lui abbiamo avuto un "dibattito" pubblico alcuni anni fa. Il mio interesse primario è cercare di ottenere la verità e non di accusare qualcuno su una giovane ragazza delle colline nere della Virginia

dell'Ovest. Inoltre, io e Linus non siamo "nemici" o niente del genere. Lo ho incontrato una volta e mi è sembrato un ragazzo simpatico, amichevole e intelligente. L'unica cosa che mi dispiace è che non ha sviluppato Linux sulla base della tecnologia a microkernel di MINIX. Con tutti i problemi di sicurezza che Windows ha adesso, è ovvio in maniera crescente a tutti che piccoli microkernel, come quelli di MINIX, sono una base migliore per i sistemi operativi che grossi sistemi monolitici. Linux è stato vittima di minori attacchi rispetto a Windows perché (1) è effettivamente più sicuro, ma anche (2) gran parte degli attaccanti prendono di mira Windows perché offre un bersaglio più grande, e quindi Windows viene attaccato di più. Come ho fatto 20 anni fa, io credo ancora fermamente che il solo modo di fare il sofware sicuro, affidabile e veloce è di farlo piccolo. Combattere le funzionalità eccessive.

Se siete arrivati fin qui, grazie per il vostro tempo. Viene dato il permesso di duplicare **questa pagina web** purché ne venga fornita la versione originale e non modificata. Viene dato analogo permesso anche per la traduzione italiana (deve essere compreso il banner nella pagina).

Andy Tanenbaum, 20 Maggio 2004

Commenti(**aggiungi il tuo**):

Giroscopio: E' chiaro che c'è dietro SCO. Una volta dimostrato che perfino Linux è stato plagiato, potranno dire in giudizio: "visto: non c'è bisogno di prove, è EVIDENTE che Linux copia Unix, sennò non potrebbe funzionare".

Killer of Daemons: Tanenbaum è un dio.

Simone: Grazie per l'articolo, molto interessante

Corrado: L'articolo è molto interessante, ma sbagli tutti i congiuntivi... Saluti

Vega: grande lavoro, con la tua traduzione mi hai fatto risparmiare parecchio tempo che avrei perso a "decodificare" lo scritto originale

Ilruz: Andy, grazie di esistere.

Prozac2000: Grazie per la traduzione che hai fatto, ho risparmiato tempo. Per quanto riguarda Linux, sono d'accordissimo sul fatto che un Kernel Monolitico non sia assolutamente adatto per soddisfare le esigenze che l'informatica odierna richiede...

XavierX: Grazie per la traduzione: sapere qualcosa in più non può che aiutarci a comprendere meglio le cose. Per me Tocqueville fino ad un minuto fa era solo una discoteca di Milano, ora è anche un pagliaccio al servizio di chi vuole affondare Linux.

Giorgio Codazzi: Premesso non ho studiato la struttura a microkernel di MINIX, non concordo con l'idea. Sul minuto d'avvio dei sistemi MS, é un colpo basso. Il confronto é più corretto se l'HW e la complessita dell'OS é simile. Ma a parte questo, credo che sia uno scontro tipo: é meglio la pizza o gli spaghetti? Credo che approcci diversi abbiano intenti diversi e diversi vantaggi. Certo, l'eleganza formale non ha sempre la meglio e questo non é molto bello. Se fosse così (infatti) il Pascal oggi dovrebbe essere al posto del C tra i linguaggi più diffusi. Però questo dice che anche tra i tecnici prevale una forza che non é puramente nella ragione. Certo, c'é anche la componente storica. Io però sono convinto che prevalga sempre smanettonia, come (in arte) al nostrano e grandissimo Canova hanno prevalso i prodotti dell'arte moderna. Questa personale teoria osserva solo che sulla ragione prevale la passione.

puukko: Sono daccordissimo sulla struttura a microkernel di MINIX. Piccoli oggetti portano sicuramente a grandi risultati, facilità di manutenzione e miglior controllo della sicurezza. Mi riporta indetro nel tempo negli anni 80 quando programmavo applicazioni commerciali con il linguaggio NL (Natural Language)della Logical Machine Corp. (Lomac) un linguaggio evoluto basato appunto su piccoli oggetti.

Pad: Dico una sciocchezza: quanto sarebbe difficile, attualmente, riscrivere il kernel di Linux in modo non-monolitico? Quanti giorni-uomo

occorrerebbero?

mauro: Grazie

Giona: Andrew Tanenbaum è secondo me il miglior "programmatore" di tutti i tempi

Daniele: Attualmente, riscrivere il kernel di linux come micro kernel, non comporterebbe solo un notevole lavoro ma implicherebbe anche la riscrittura di tutti i driver e le componenti dato che oltre a cambiare l'architetura queste vanno a finire in user mode

Allegati:

vote.jpg

Contatto: michele at sciabarra dot com