Linux kernel

Una breve introduzione al cuore del sistema operativo GNU/Linux.

28 ottobre - Linux Day 2006 - GLUG

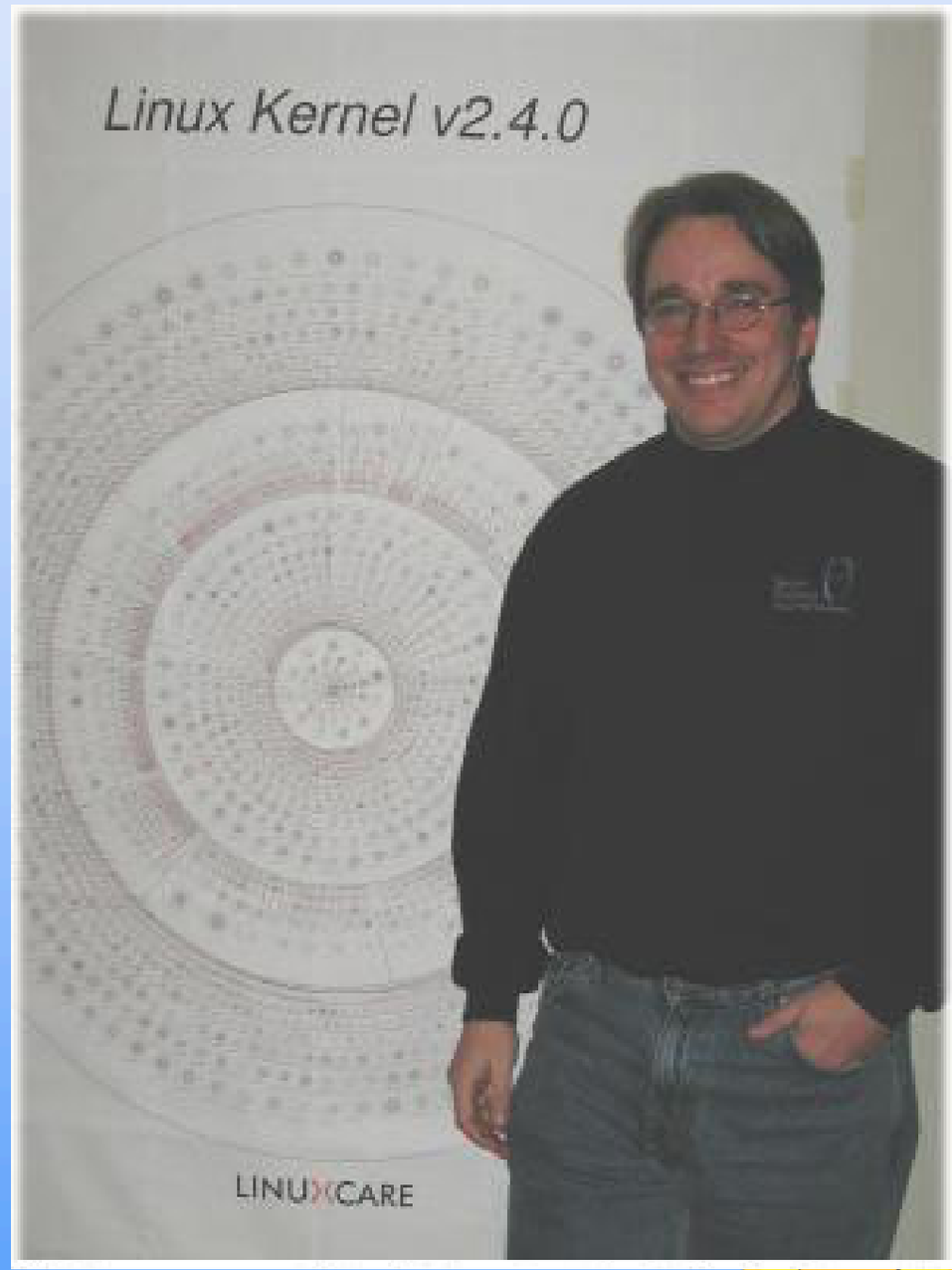
Daniele Venzano

Indice

- Introduzione
- I componenti di un sistema operativo
- I compiti di un kernel
  - Come li realizza Linux
  - I sottosistemi principali
- Come viene sviluppato Linux
  - Modello di sviluppo estremamente distribuito
  - E’ possibile dare una mano ?

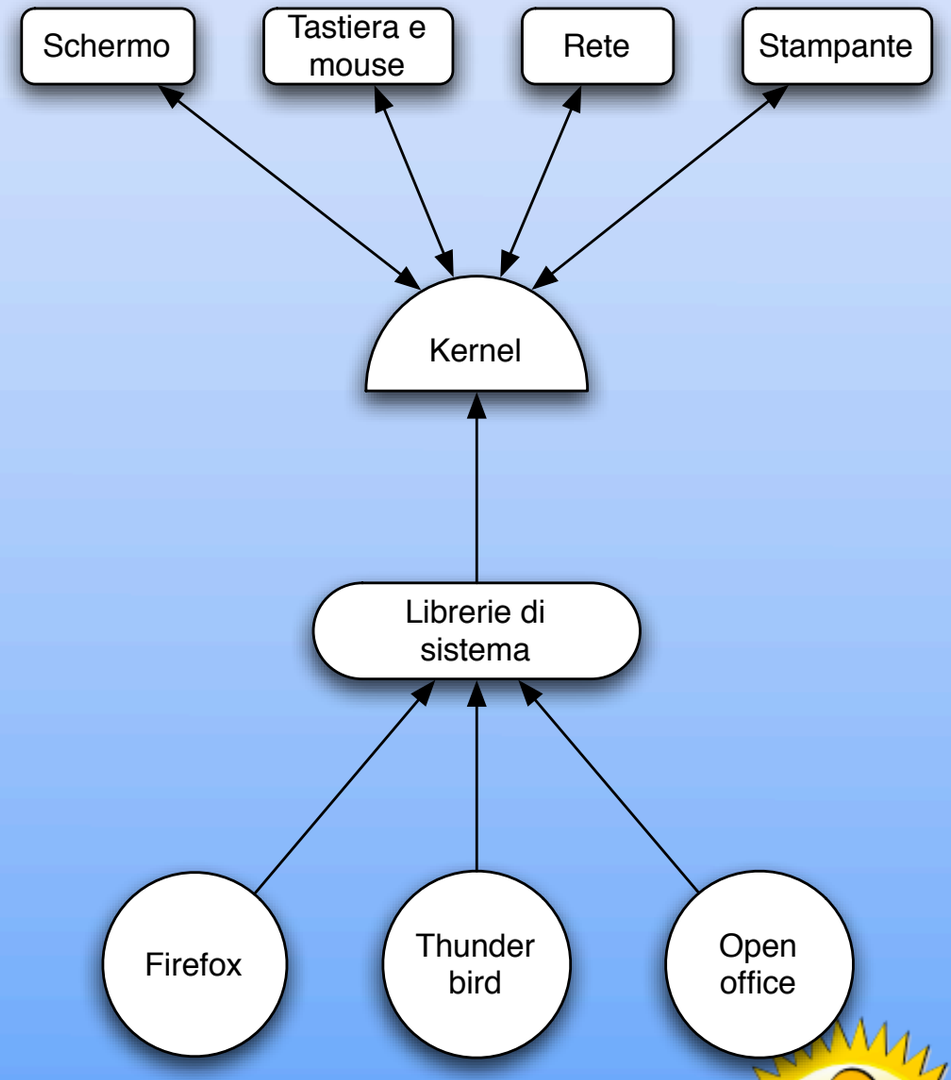
Introduzione

- Linux è nato nel 1991 grazie alla pigrizia di Torvalds ed al freddo e buio inverno finlandese
- Ad oggi supporta 24 architetture differenti
- Centinaia di sviluppatori ci lavorano ogni giorno
- Quest'anno Linux compie 15 anni



Sistema operativo

- Il software che viene eseguito da un computer può essere suddiviso in:
  - kernel
  - librerie di sistema
  - applicativi utente (browser, posta elettronica, ...)



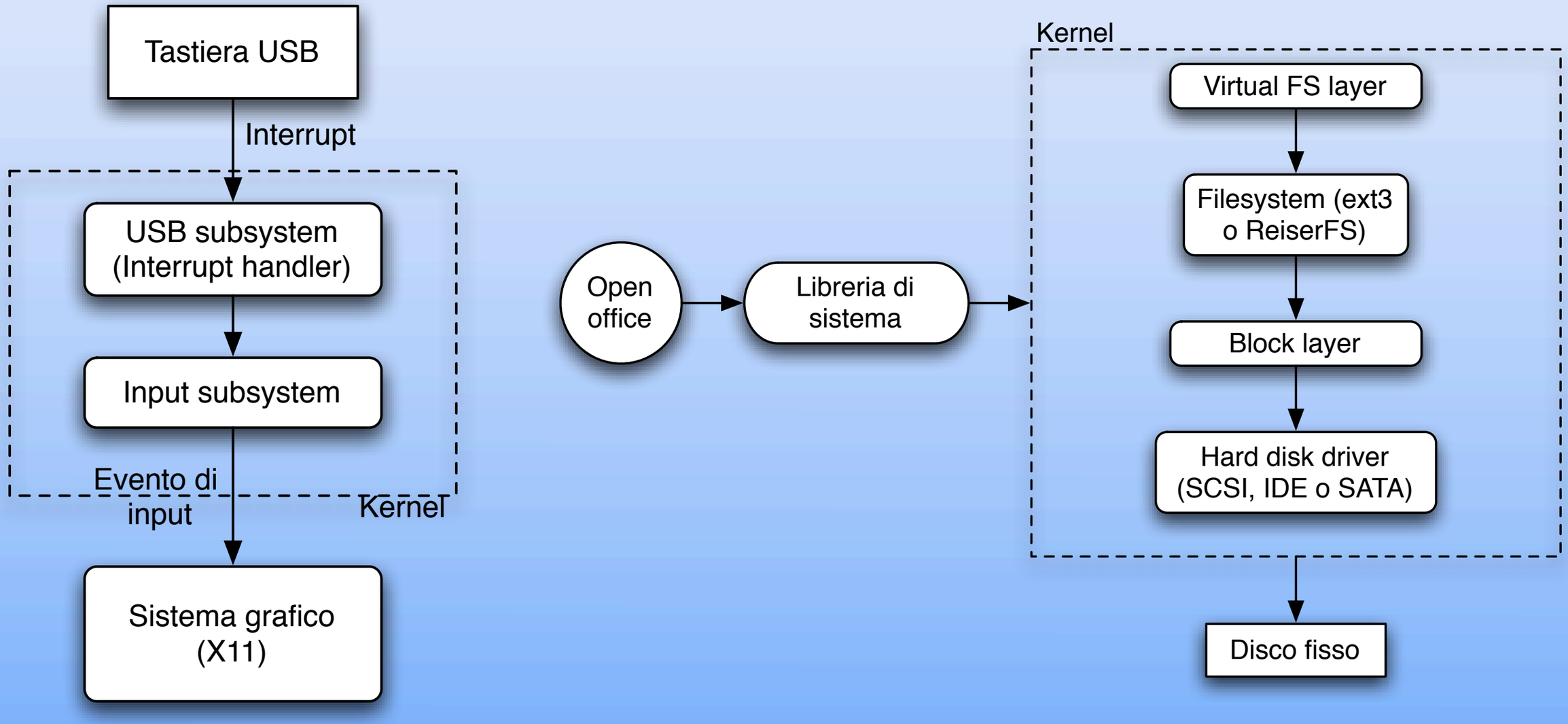
Kernel

- Il kernel si occupa di:
  - consentire l'esecuzione di più programmi nello stesso tempo (multitasking, multithreading)
  - consentire l'accesso a più utenti (anche contemporaneamente)
  - fornire un accesso trasparente all'hardware
  - essere robusto e sicuro in tutte queste cose

Kernel

- Linux è organizzato in diversi sottosistemi, ognuno che si occupa di un compito specifico e ben delimitato. Alcuni esempi sono:
  - USB
  - Input (tastiere, mouse, touchscreen, penne ottiche, ...)
  - Gestione della memoria
- Ogni sottosistema gestisce diversi driver, per la comunicazione con le componenti hardware o per realizzare determinati compiti
- Un driver può appartenere a più sottosistemi (es. tastiera USB)
- I driver sono distribuiti sotto forma di uno o più “moduli”

Kernel



Pressione di un tasto

Apertura di un file con OpenOffice

Sviluppo

- Il kernel è sviluppato interamente in linguaggio C con dei piccoli pezzetti in assembler eseguiti durante il boot
- Torvalds mantiene un albero dei sorgenti che rilascia ogni 6/8 settimane e che viene considerato ‘stabile’
- Esistono un certo numero di altri alberi di ‘sviluppo’ in cui vengono preparate le modifiche da includere nell’albero stabile
- Spesso ogni albero di sviluppo corrisponde ad un sottosistema ed è gestito da un ‘subsystem maintainer’.

Modello di sviluppo

- Torvalds decide quali modifiche applicare e quali scartare
- Ogni sottosistema ha un “subsystem maintainer” che si occupa di integrare le modifiche specifiche al suo sottosistema e di girarle a Torvalds quando lo ritiene opportuno
- Quasi tutti i driver hanno uno o più “driver maintainer”
- Poi ci sono anche:
  - collaboratori occasionali
  - sviluppatori “out of tree”, spesso a contratto di produttori di hardware

Modello di sviluppo

Modello di sviluppo

- Tutte le modifiche al kernel vengono diffuse sotto forma di patch
- Recentemente è stato messo a punto un sistema di tracciamento che permette di seguire il percorso di ogni patch, da chi l'ha creata a chi l'ha inclusa nell'albero di Torvalds
- Tutte le patch più grosse vengono sottoposte a diversi cicli di revisione prima di essere incluse. Vengono controllati:
  - correttezza
  - forma del codice sorgente (CodingStyle)
  - forza di carattere di chi invia la modifica

Dare una mano

- Il codice sorgente scritto per il kernel ha requisiti molto differenti da quelli per le applicazioni normali
- E' possibile iniziare a 'capire come funzionano le cose' frequentando il sito kernelnewbies.org, dove si può trovare una mailing list ed una canale chat per chiedere informazioni
- Su kerneljanitors.org c'è un elenco di cose da fare, utili per prendere confidenza con i diversi sottosistemi
- Su bugzilla.kernel.org c'è il sistema di tracciamento dei bug di Linux.

Links

- Kernel Newbies (aiuto per principianti)
- Kernel Janitors (lavori di pulizia e manutenzione)
- Kernel bugzilla (tracciamento dei bug)
- Kernel map
- Kernel geomap
- Linux kernel swear counter