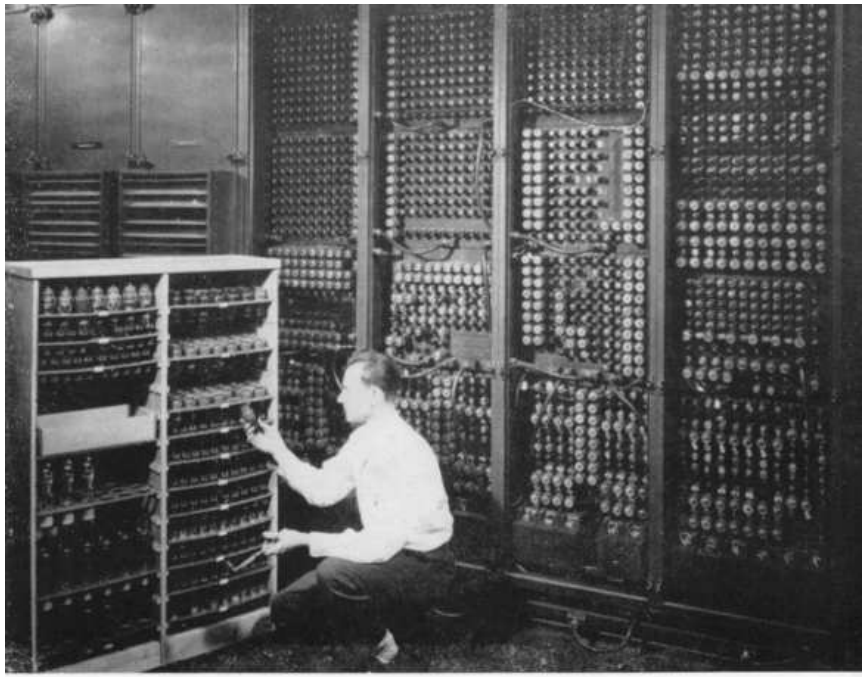
2. Einführung

2.1 Geschichte

Erste Generation: 1945-1955

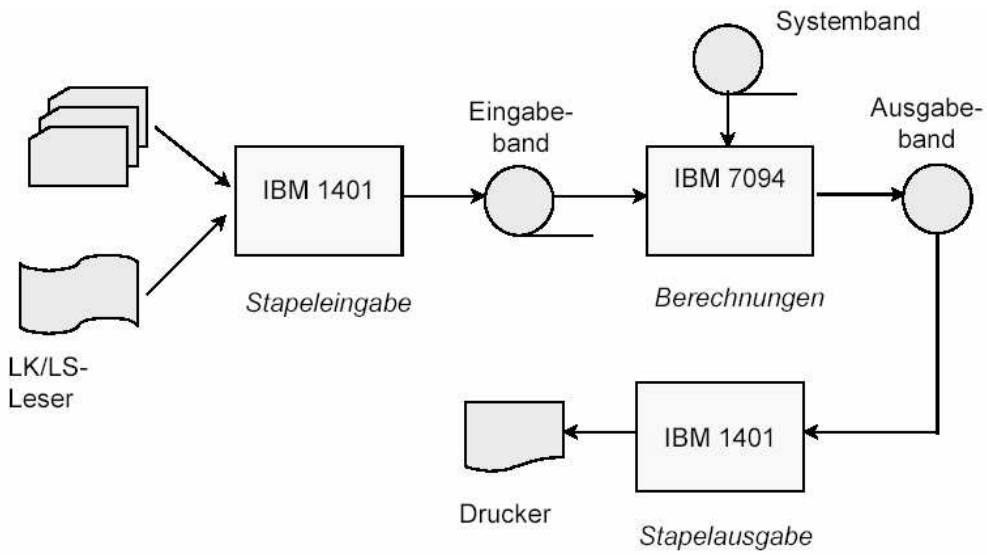
- **Röhren** und Steuerung durch **Klinkenfelder**.
- K. Zuse (irb.cs.tu-berlin.de/~zuse/Konrad_Zuse).
- Fest verdrahtete Programme
  → kein Betriebssystem.
- Aufwendiger Betrieb und Wartung.
- Für numerische Berechnungen
  (z.B. Tabellen für Logarithmus).
- ENIAC (1946):
  - **E**lectronic **N**umerical **I**ntegrator **A**nd **C**omputer,
  - ~19.000 Röhren,
  - 27.000 kg, …



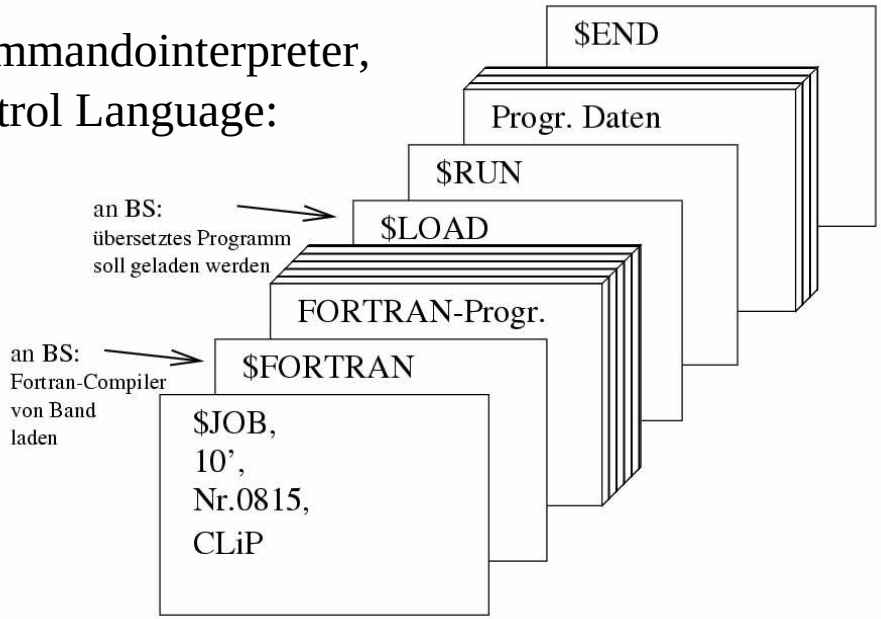
Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

Zweite Generation: 1955-1965

- Erhöhte Zuverlässigkeit durch **Transistoren**.
- Aber sehr teuer (mehrere Millionen EUR).
- Programme nun auf Lochkarten:
  - gelochtes maschinenlesbares Papier,
  - Sprachen: Assembler, Cobol und Fortran.
- Anwendungen: wiss. & techn. Berechnungen.
- **Stapelbetrieb** (batch processing):
  - **Job** = Programm oder Menge von Prgs.
  - **Stapel** = Sammlung von Jobs.
  - sequentielle Verarbeitung der Programme.
  - **keine Interaktion** zw. Prg. & Benutzer.
  - manuell: Operateur lädt neuen Job, wenn der Alte abgearbeitet ist → ineffizient.
  - automatisch: kleiner, billiger Rechner liest neue Jobs und speichert diese auf Band. Dieses Band dient als Eingabe für den Mainframe (siehe Bild).



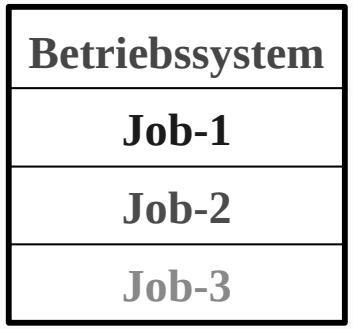
- **Monitor**:
  - = speicherresidenter Kommandointerpreter,
  - gesteuert durch Job Control Language:



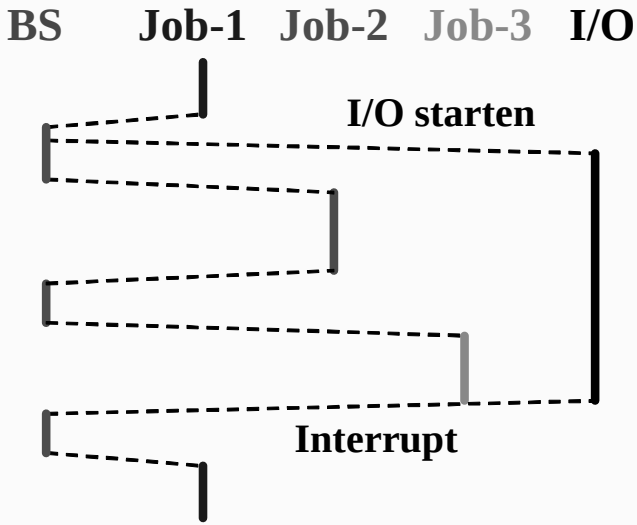
- Betriebssystem, Beispiel IBSYS für IBM 7094:

Dritte Generation: 1965-1980

- Einführung von **integrierten Schaltungen**.
- Zunehmend alphanumerische Datenverarbeitung.
- IBM System/360:
  - Serie von kompatiblen Rechnern (1964).
  - Stapelbetrieb für wissenschaftlichen und kommerziellen Bereich.
  - OS/360 sehr komplex:
    - mehrere Mio. Zeilen Assembler-Code,
    - >1000 Entwickler,
    - viele Bugs.
- **Mehrprogrammbetrieb** (Multiprogramming):
  - mehrere Jobs im Hauptspeicher,
  - Verteilen der CPU nach Prioritäten,
  - Und E/A-Wartezeiten ausnutzen. ➔



Speicher



- Zusätzlich **Spooling**-Technik eingeführt:
  - **S**imultaneous **P**eripheral **O**peration **O**n **L**ine,
  - Jobs von Lochkarten einlesen & zw.speichern.
  - Nachladen eines Jobs, sobald Speicher frei.
  - analog für Job-Ausgabe.
  - heute: Ansteuerung von Drucker.
- Probl.: kein interaktiver Dialog bei Stapelbetrieb & Multiprogramming.
- Lösung: **Time Sharing** (Zeitscheibenkonzept):
  - Mehrprogrammbetrieb mit interaktiver Nutzung.
  - Dialogbetrieb: jeder Benutzer ist über ein Terminal mit dem System verbunden.
  - quasi-parallele Ausführung von Jobs → **Multitasking**
  - Ablauf:
    - Job erhält CPU für best. Zeitscheibe Δt.
    - wenn Job in Δt nicht fertig, wieder in Warteschlange für CPU einreihen.
  - kurze Zeitscheiben zur Unterstützung interaktiver Benutzer.
  - CTSS (Conventional Time Sharing System)
    → erstes Time Sharing System vom MIT (Boston, USA, 1962).
- Oft Kombination von:
  - Hintergrundarbeit (Batch Processing) und
  - Vordergrundarbeit (Time Sharing).

Geschichte von Unix:

- Vorläufer: MULTICS, 1965:
  - **Mult**iplexed **I**nformation and **C**omputing **S**ystem.
  - Idee aus Stromversorgung: bei Bedarf steckt man Stecker in Dose u. bezieht Elektrizität.
  - von MIT, Bell Labs, General Electrics.
  - wurde nicht kommerziell genutzt.
  - aber viele neue Ideen: Mehrbenutzer, Ring-Protection, File-Mapping, Prozesse, …
- Ken Thompson schrieb abgespeckte Version: UNICS (**Uni**plexed **I**nformation and **C**omputing **S**ervice) für Einbenutzersysteme.
- UNIX (1969):
  - Dennis Ritchie entwickelte die Sprache C.
  - schrieb UNICS zusammen mit Thompson neu in C → UNIX
  - 8.200 Zeilen C und 900 Zeilen Ass.
  - nachfolgend vielfach portiert und erweitert.
- Minix (1987):
  - Andrew S. Tanenbaum,
  - Mikrokern-Architektur,
  - wurde im Lehrbetrieb eingesetzt,
  - war schnell beliebt bei den Studenten, die das System anschließend erweiterten.

- Linux (1991):
  - Linus Torvalds (finnischer Informatikstudent),
  - schreibt Minix von Grund auf neu:
    - monolithische Architektur
    - speziell für i386 Plattform.
  - unter GNU Public License.
- Gründung der SUSE GmbH (1992).
- Linux Kernel Version 1.0 (1994).
- AMD64 Versionen verfügbar.
- Dennis Ritchie (stehend) & Ken Thompson vor einem PDP-11 System →

Quelle: http://www.psych.usyd.edu.au/pdp-11/real_programmers.html

Vierte Generation: 1980-1990

- Hardware:
  - large scale integrated circuits (LSI).
  - Bauelemente mit 1000en von Transistoren.
  - Entwicklung von Mikrocomputern und PCs.
- Software:
  - BS überwinden Rechnergrenzen (IP-Protokolle).
  - graphische Benutzschnittstellen.
  - Parallelität durch Threads.
- Die Geschichte wiederholt sich:
  - zuerst einfache Stapel-Systeme, zum Beispiel CP/M (= Control Program for Microcomputers).
  - bis zu multitaskingfähigen Betriebssystemen, z.B. Unix & NT.
- Betriebssysteme:
  - CP/M (8-Bit) für Intel 8080.
  - MS-DOS (16-Bit) für Intel 80x86.
  - Microsoft Windows (16/32-Bit) für 80x86.
  - Unix (32-Bit) für Workstations (X-Window).
  - MacOS: Einführung einer graphischen Benutzeroberfläche.

- Netzwerkbetriebssysteme:
  - Def.: Ein Netzwerk-BS beinhaltet Funktionen zur Kommunikation mit anderen Rechnern.
  - Benutzer haben über Netzwerk Zugang zu anderen Rechnern.
  - Beispiele: Novell, Windows NT, Unix.
- Verteilte Betriebssysteme:
  - Def.: Ein verteiltes Betriebssystem stellt verteilte Betriebsmittel in einer für den Benutzer transparenten Art und Weise zur Verfügung.
  - Beispiele: Amoeba, Mach, Chorus,....

Geschichte von Windows NT:

- IBM und Microsoft entwickeln ab 1988 zusammen OS/2.
- Kooperation scheitert endgültig 1991:
  - Bill Gates beginnt mit Windows NT:
  - Leiter: D. Cutler maßgeblich an VMS beteiligt.
    → viele Gemeinsamkeiten zw. VMS & NT.
  - mehrjährige Entwicklung mit ca. 200 Programmierern
- Windows NT 3.1 (Ende 1993):
  - 32-Bit System für Alpha, PowerPC, Intel MIPS.
  - Unterstützung für alte 16-Bit DOS/Windows Programme.

- Windows 2000 (2000):
  - USB, FAT32, PnP, Power-Management.
  - nur noch x86 CPUs unterstützt.
- Windows XP (2001):
  - XP = Experience.
  - bessere Unterstützung für Spiele.
  - führt 98/ME und NT/2000 zusammen.
- Windows XP x64 Edition: für AMD64 (Beta verfügbar).

Moderne Entwicklungen

- Cluster Systeme:
  - Hochverfügbarkeit durch Redundanz.
  - Bündelung von Rechenleistung.
  - Rechner im LAN vernetzt.
- Grid Systeme:
  - massiv paralleles System bestehend aus u.U. weit entfernten heterogenen Rechnern.
  - Aspekte: Anbieten, Auffinden und die Nutzung von Ressourcen im Netz.
  - Idee entstammt Stromversorgung (Power Grid) bei Bedarf steckt man Stecker in Dose u. bezieht Elektrizität.
  - Bem.: gleiche Idee wie bei MULTICS.

- Eingebettete Systeme:
  - maßgeschneiderte kleine Betriebssysteme
    - z.B. OSEK im KFZ-Bereich
    - TinyOS für Sensornetzwerke.
  - typischerweise mit Echtzeitanforderungen.
- Multimedia:
  - Unterstützung von Audio- und Videoströmen.
  - Dienstegarantien (Quality of Service).
- Interoperabilität: Unterstützung heterogener Umgebungen.

2.2 Arten von Betriebssystemen

- **Mainframe (Großrechner) Betriebssysteme**:
  - früher in großen Rechenzentren, heute für Web-/E-commerce-Server, …
  - große Ein- und Ausgabebandbreiten:
    - mögl. viele Prozesse effizient ausführen
    - z.B. 15.000 aktive Benutzer
  - unterschiedliche Betriebsarten:
    - Stapelbetrieb, z.B. Schadensmeldungen in einer Versicherung
    - Timesharingbetrieb bei zentralen Diensten
    - Transaktionsb., z.B. Banküberweisungen, Flugbuchungen, …
  - Zugang über einfaches Terminal.
  - Vorteil: zentrale Administrierung
  - Nachteil: hoher Preis der Hardware.
  - Beispielsystem IBM OS/390 (MVS Nachfolger).
- **Server Betriebssysteme**:
  - eine Ebene unterhalb der Mainframes.
  - bedienen viele Nutzer über ein Netzwerk.
  - vgl. wenig interaktive Benutzer: ca. 10-100.
  - verteilen Hardware- und Software-Ressourcen.
  - für Datei-, Mail-, Web-, Druckserver, …
  - Beispiele: Unix, Linux, Windows Server 2000/2003.

- Netzwerk-Betriebssysteme:
  - verwaltet gemeinsame Ressourcen im Netz.
  - Benutzer können sich an fremden Maschinen anmelden, um Dienste in Anspruch zu nehmen.
  - Beispiel: Novell Netware, Windows XP Home/Professional.
- **Verteilte Betriebssysteme**:
  - Rechnerverbund bleibt Benutzer verborgen.
  - Wo die Programme ausgeführt werden und wo die benötigten Daten liegen wird durch das BS bestimmt und ist dem Benutzer nicht bekannt.
  - Benutzer hat den Eindruck einer einheitlichen Computerressource.
  - Beispiele: Amoeba, Kerrighed, Plurix.
- PC-Betriebssysteme:
  - beschränken sich auf einen Benutzer.
  - benutzerfreundliche Oberfläche.
  - einfache Installation und Wartung.
  - Unterstützung vieler Peripheriegeräte.
- **Echtzeit-Betriebssysteme**:
  - Verarbeitung von Daten innerhalb bestimmter, üblicherweise enger Zeitgrenzen.
  - Je nach Toleranz bzgl. Überschreitung von Sollzeitpunkten wird zwischen harter und weicher Echtzeit unterschieden.
  - Für Überwachungs-, Fertigungsanlagen, …
  - Beispielsysteme: VxWorks, QNX, ...

- Betriebssysteme für eingebettete Systeme:
  - besitzen oft Echtzeiteigenschaften.
  - optimiert bzgl. Programmgröße, verfügbarem Arbeitsspeicher und Stromverbrauch.
  - typischerweise für Mikrocontroller oder PDAs.
  - Beispiele: OSEK, TinyOS, PalmOS, Windows CE, ...
- Betriebssysteme für Chipkarten:
  - für Geldkarten, Krankenversicherung, ...
  - harte Bed. an Speicher und Rechenleistung.
  - oft nur einzelne Funktionen, wie beispielsweise elektronisches Bezahlen, realisiert.
  - SmartCards gewinnen zusehens an Bedeutung.
  - Java-orientierte SmartCards mit JVM im ROM-Speicher vorhanden.
  - hohe Sicherheitsanforderungen.

2.3.1 Betriebsmittelzuteilung

- Betriebsmittel werden vom System zugeteilt:
  - um Doppelbelegungen zu vermeiden,
  - zwecks Rückgabe beim Programmabsturz.
- Zuteilung jeweils an ein "Objekt", z.B. an:
  - Programm, Prozess, Thread, Instanz, User ...
- Hauptspeicher wird benötigt für:
  - Prozeduraktivierungen auf dem Keller,
  - Puffer für Dateien und Datenpakete,
  - Codesegmente,
  - Heap-Objekte.
- Geräte / Devices:
  - entweder nur exklusiv nutzbar oder gleichzeitig nutzbar (sharable),
- Plattenspeicher wird als Dateien vergeben.
- Bei Bedarf wird der Prozessor zw. verschied. Threads umgeschaltet.
- Bildschirmoberfläche ist kostbar → Fenster.

2.3.2 Bedienerschnittstelle

- Anzeigen von Fehlersituationen.
- Dialogsteuerung für:
  - Navigation im (Datei-)System:
  - Anwenderprogramme.
  - Systembedienung.
- Einheitliche Bedienung verschiedener Programme.

2.3.3 Abstraktion von der Hardware

Für den Anwendungsprogrammierer:

- Subset von Laufzeitroutinen ist allen Unix Systemen gemeinsam:
  - create, open, close, read, write,
  - malloc, fork, wait, waitpid,
  - socket, bind, printf, scanf ...
- Viele Karten unterstützen Grafikstandards:
  - DirectX, OpenGL (, QuickDraw).
- Java Programme laufen auf beinahe jedem Rechner, wenn dort eine Java Virtual Machine (JVM) vorhanden ist.

Für den Systemprogrammierer:

- Wenn eine "Hardware Abstraktionsschicht" eingezogen wird, ist es leichter, ein Betriebssystem auf eine neue Rechnerplattform zu bringen.
- Erhalten bleiben etwa:
  - Formate im Dateisystem,
  - Protokolle auf dem Netz,
  - Algorithmen und Strategien ...

2.3.4 Verzeichnisdienste

- "Machines use addresses, people prefer names".
- Übersetzung zwischen Namen und Adressen.
- Adressen für:
  - Hauptspeicher, Ethernet, Internet, Disksektoren, E/A-Ports, SCSI-Devices ...
- Suchpfade für Klassen & Bibliotheksmodule.
- Navigation im Namensraum.
- Dateiverzeichnisse:
  - Dateinamen auf Diskadressen abbilden,
  - Aliases, Shortcuts und Decknamen,
  - geschachtelte Unterverzeichnisse.
- Registry in Windows:
  - hierarchische Datenbank; speichert Konfiguration, Schlüssel und Werte.
- Auffinden von Namen im Netz:
  - Namen für Benutzer & Dienste, Maschinennamen.
- Namen für Objekte:
  - Fenster und Dialogelemente, Prozesse, Ereignisse, Speicherabschnitte, ...

2.3.5 Fehlerbehandlung & Zugriffsschutz

- "Post Mortem" müssen die Betriebsmittel eines Programms wieder freigegeben werden:
  - Hauptspeicher,
  - offene Dateien,
  - Peripheriegeräte,
  - Bildschirmfenster,
  - Warteschlangenpositionen ...
- Der Benutzer wünscht sich eine verständliche Fehlermeldung:
  - kein Hex-Dump,
  - kein blauer Bildschirm,
  - keine allgemeine Schutzverletzung.
- Schutz vor:
  - unabsichtlicher Beschädigung des System,
  - böswilligen Einbruchsversuchen,
  - Monopolisierung des Systems,
  - Fehlern im Betriebssystem,
  - Hardwaredefekten.

2.3.6 Konfigurierung & Startup

- Konfigurierungsbedarf für:
  - Benutzerrechte, Festplatteneinteilung,
  - Netzwerkprotokolle, Bildschirmauflösung, BIOS-Einstellungen ...
- Automatische Erkennung von Hardware.
  - Plug & Play vs. Plug & Work,
  - für Geräte am PCI Bus, für ISA Geräte?
- Initialisierung:
  - der Hardwarekomponenten,
  - des Betriebssystemkernes,
  - der Serverdienste.
- Ausschalten des Systems:
  - Dokumente sichern lassen,
  - Dateipuffer auf Disk schreiben,
  - Druckerwarteschlange schließen,
  - Abmelden am Netz, Services beenden, Power down !
- Für mobile Geräte: Schlafmodus einleiten.