

## 7 Benutzungsschnittstellen

Klassischer Zugriff auf einen Host-Rechner („Großrechner“, „Mainframe“, „Server“)

- Rechner ist im Rechnerraum, Zugriff über „Textterminals“
- Textterminals (ASCII-Terminals, vt100-Terminals, IBM 3270) bestehen aus
  - o Bildschirm zur Anzeige von Buchstaben (80x24, monochrom, nicht grafikfähig)
  - o Tastatur
  - o Mini-CPU
  - o Serielle Verbindung zum Rechner
- Üblicherweise 1-100 (oder sogar 1000) Terminals am Rechner
- Login-Bildschirm bei der Anmeldung, danach Kommandozeile
- Massiv Multiuserfähig.

Workstation / PC / Laptop

- Rechner mit einem Benutzer
- Grafikfähige Anzeige, Ton, Multimedia, Grafikkarte (KVM = Keyboard, Video, Mouse)
- Graphische Benutzungsoberfläche (GUI)
- Üblicherweise nur ein Benutzer am Rechner (PC = „Persönlicher Computer“)
- Zunehmend Einsatz im Multiuser/Serverbetrieb („headless“) ähnlich dem Host

Thin Clients

- Wie PC, jedoch ohne Festplatte („diskless“) und vergleichsweise leistungsarme CPU
- Rechner dient nur der Ein/Ausgabe, Rechenleistung auf Server

### GUI vs. Kommandozeile

- Skriptfähigkeit
- Flexibilität
- Erlernbarkeit
- Remote-Zugriff
- Multiuserfähigkeit

Kombination aus beidem ist meist sinnvoll.

Situation heute?

Textterminal und GUI sind netzwerkfähig, d.h. der Rechner an dem man sitzt ist nicht notwendigerweise der, an dem man „arbeitet“, d.h. auf dem die Prozesse laufen, die man startet. Rechnerräume wichtiger denn je.

UNIX-Philosophie ist eng verwoben mit der Kommandozeile:

- „small is beautiful“
- Jedes Programm sollte eine Sache machen, die aber wirklich gut
- Speichere Daten in Textdateien
- Programme sollten wieder verwendbar sein
- Alles - auch Hardware - ist eine Datei

Klassisch: Rechner startet mit Kommandozeile, GUI wird als „Programm“ aufgerufen. Windows hat nur schwache (DOS-basierte) Kommandozeile. Zukunft: MSH

### **Zugriff auf einen Rechner via Netz:**

Fileserver, Webserver, Mailserver, ...

Auf dem Server läuft ein daemon/Dienst, der auf Socket-Verbindung wartet. Anfrage an Dienst gemäß Protokoll, Dienst antwortet. Basis: Sockets und TCP/IP

Vordefinierte Ports für Dienste (Webserver: 80, Mailserver: 25 (senden), 110 (empfangen)). Dienste mit Portnummer < 1024 können nur vom Benutzer „root“ angeboten werden.

Dienst kann verstanden werden als „entfernter Prozeduraufruf“. Besondere Dienste, die genau dies umsetzen und damit „programmierbar“ sind: RPC, CORBA, Webservices (XML, „lesbar“)

Idee: Parameter „einpacken“, übertragen, auspacken. Entfernter Rechner für Prozedur aus, packt Ergebnis ein, überträgt und lokaler Rechner packt Ergebnis aus.

Beschreibung der API sowie Repräsentation der Datenstrukturen meist von Programmiersprachen und Betriebssystem unabhängig.

### **Dienste mit Zugriff auf das Userinterface**

Das Netzwerk-Textterminal: telnet/ssh

- extrem niedrige Bandbreite, Zugriff quasi immer möglich. Standard bei Unix.
- Volle Kontrolle über den Rechner und sämtliche Dienste
- Rechner im Rechnerraum, Entwickler im Büro

- KEIN Fileserver, sondern entferntes Textterminal
- Port 22/23

#### Netzwerk-GUI: X11

- X11 als Protokoll zur Kommunikation zwischen Client (Programm) und Server (Prozess, der Grafikkarte, Tastatur, Maus) steuert
- Kommunikation zw. Client und Server entweder shared memory (beide auf selbem Rechner) oder Sockets.
- Relativ niedrige Bandbreite
- Quasistandard in der Unix-Welt (KDE, GNOME basierend auf X11)
- Port 6000-

#### Netzwerk-GUI: Kopieren von Bildschirminhalten (VNC, RDP, Citrix, ...)

- Entfernter Rechner baut virtuellen oder realen Bildschirm auf
- Bildschirminhalte werden über Netz kopiert
- Recht hohe Bandbreite
- (inzwischen) Standard bei Windows
- Problem: nicht Multiuserfähig, da der entfernte Bildschirm „kopiert“ wird.

### **Verzeichnisdienste**

Wo liegen die Daten des Benutzers?

- Auf (isoliertem) Rechner
- Auf Server. Anmelden mit gleicher Kennung damit auf mehreren Rechnern möglich. Häufig identische Arbeitsumgebung auf unterschiedlichen Rechnern.
  - o Active Directory (Windows)
  - o NIS+, Netinfo, LDAP (Unix)

## Was haben wir nicht gemacht?

Sicherheit

- Buffer Overflows, DoS
- (Bootsektor-)viren
- Kryptographie

Eingabe/Ausgabe

BIOS, Bootvorgang, Treiber

Shared Libraries (dll, .so)

Softwareinstallation, Abhängigkeits- und Versionsmanagement

Powermanagement

Mehrprozessorsysteme

Verteilte Betriebssysteme

## URLs für interessierte:

[www.freebsd.org](http://www.freebsd.org)

[www.openbsd.org](http://www.openbsd.org)

[www.netbsd.org](http://www.netbsd.org)

<http://www.apple.com/macosx/>

<http://www.kernel.org/>

<http://www.sun.com/software/solaris/>

<http://www-1.ibm.com/servers/aix/>

<http://www.microsoft.com/windows/default.msp>