

I. Informationssysteme

Kapitel 2: Modelle und Architekturen

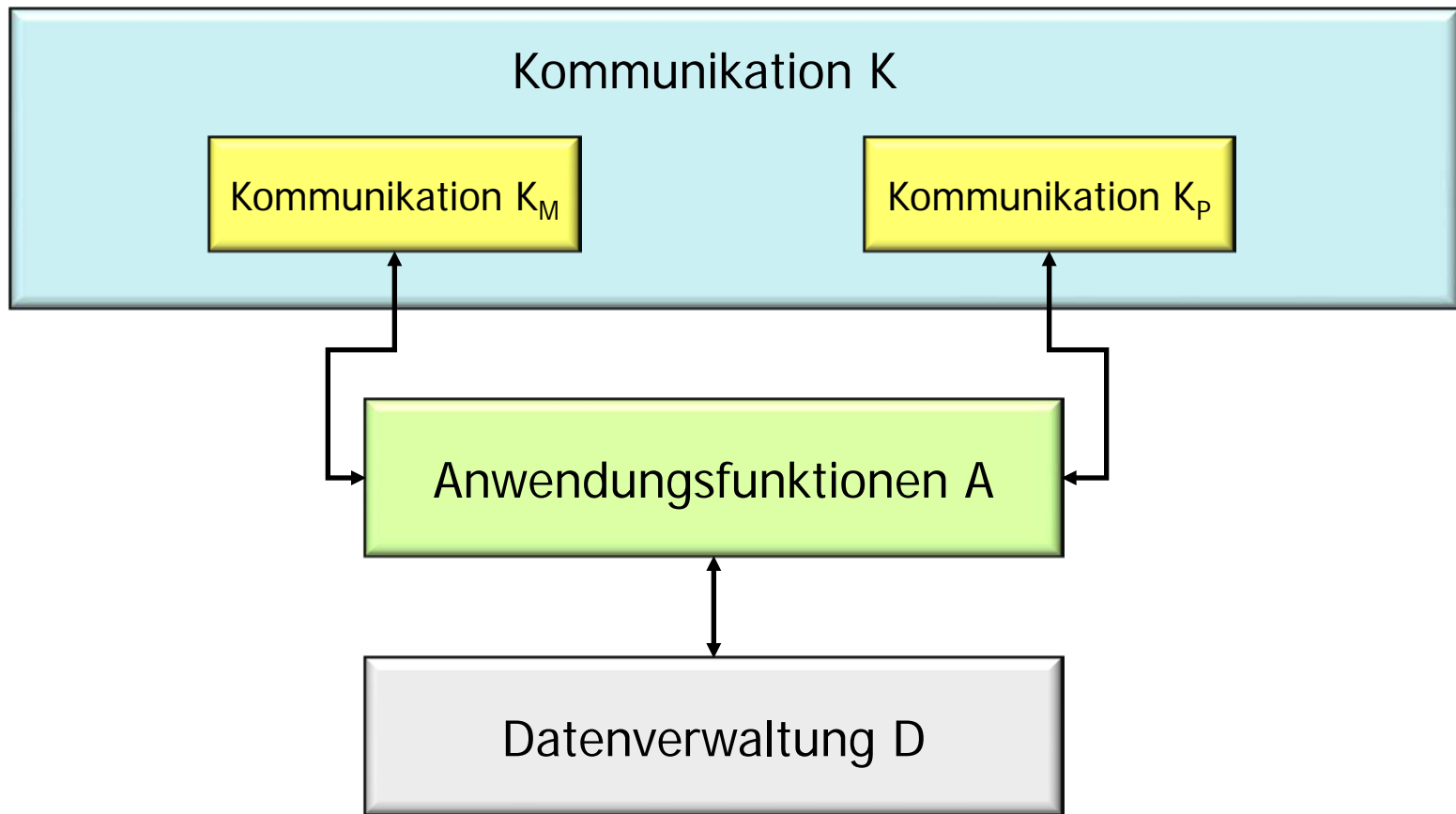
- 1. Modelle für die Architektur von Informationssystemen
- 2. Architekturansätze für Informationssysteme

Anforderungen

- Minimierung der Komplexität
- Skalierbarkeit
- Portierbarkeit
- Wartbarkeit
- Standardisierung
- Wohl-definierte Schnittstellen
- Unabhängigkeit

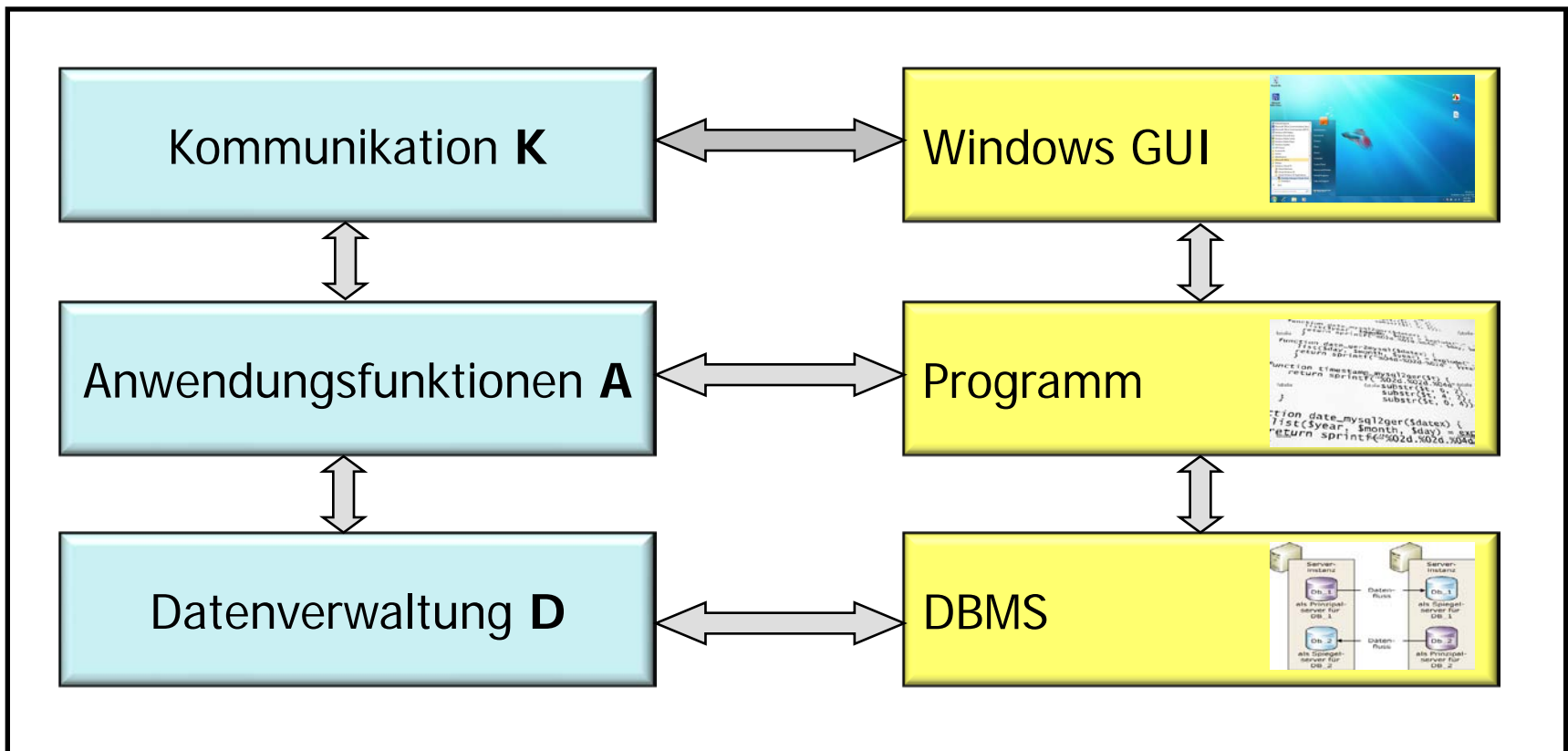
Modularisierung
der IS Architektur

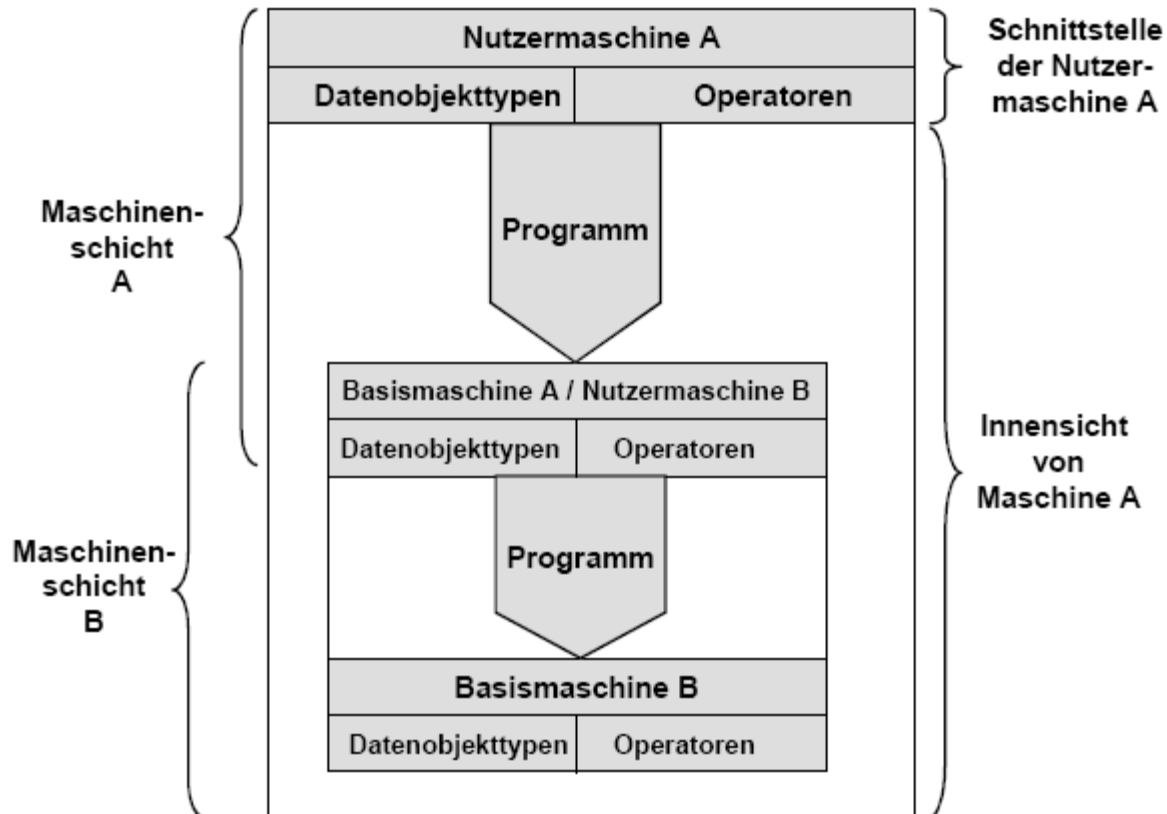
- **ADK-Strukturmodell**
Gliedert ein Informationssystem in die Teilsysteme Anwendungsfunktionen (A), Datenverwaltung (D) und Kommunikation (K). Letztere wird weiter in Kommunikation mit Personen (K_p) und Kommunikation mit Maschinen (K_M) unterteilt.
- **Nutzer- und Basismaschinen Modell**
Beschreibt die Struktur einer programmgesteuerten Maschine aus der Innen- (Basismaschine & Programm) und Außenansicht (Nutzermaschine).



K_P : Kommunikation mit Personen
 K_M : Kommunikation mit Maschinen

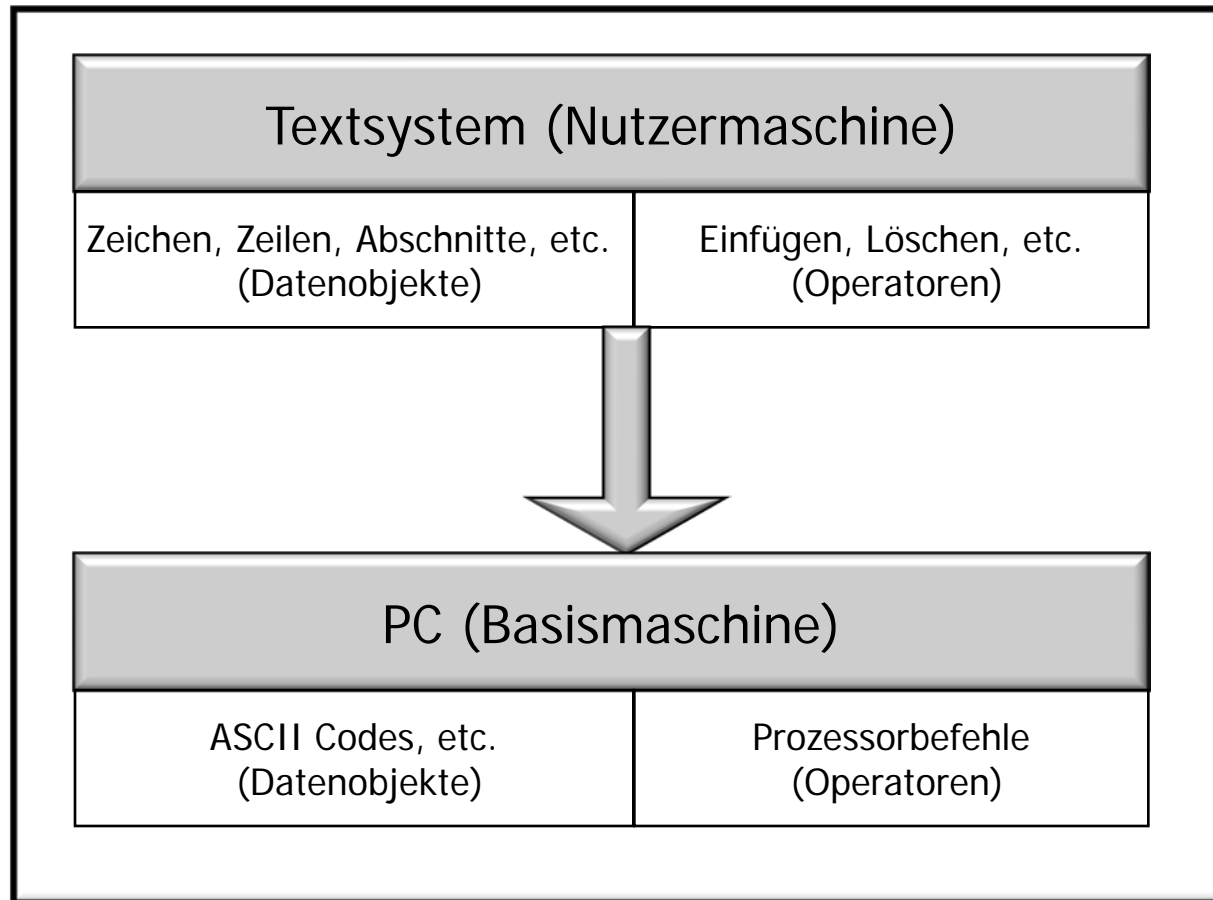
Konventionelles Anwendungssystem

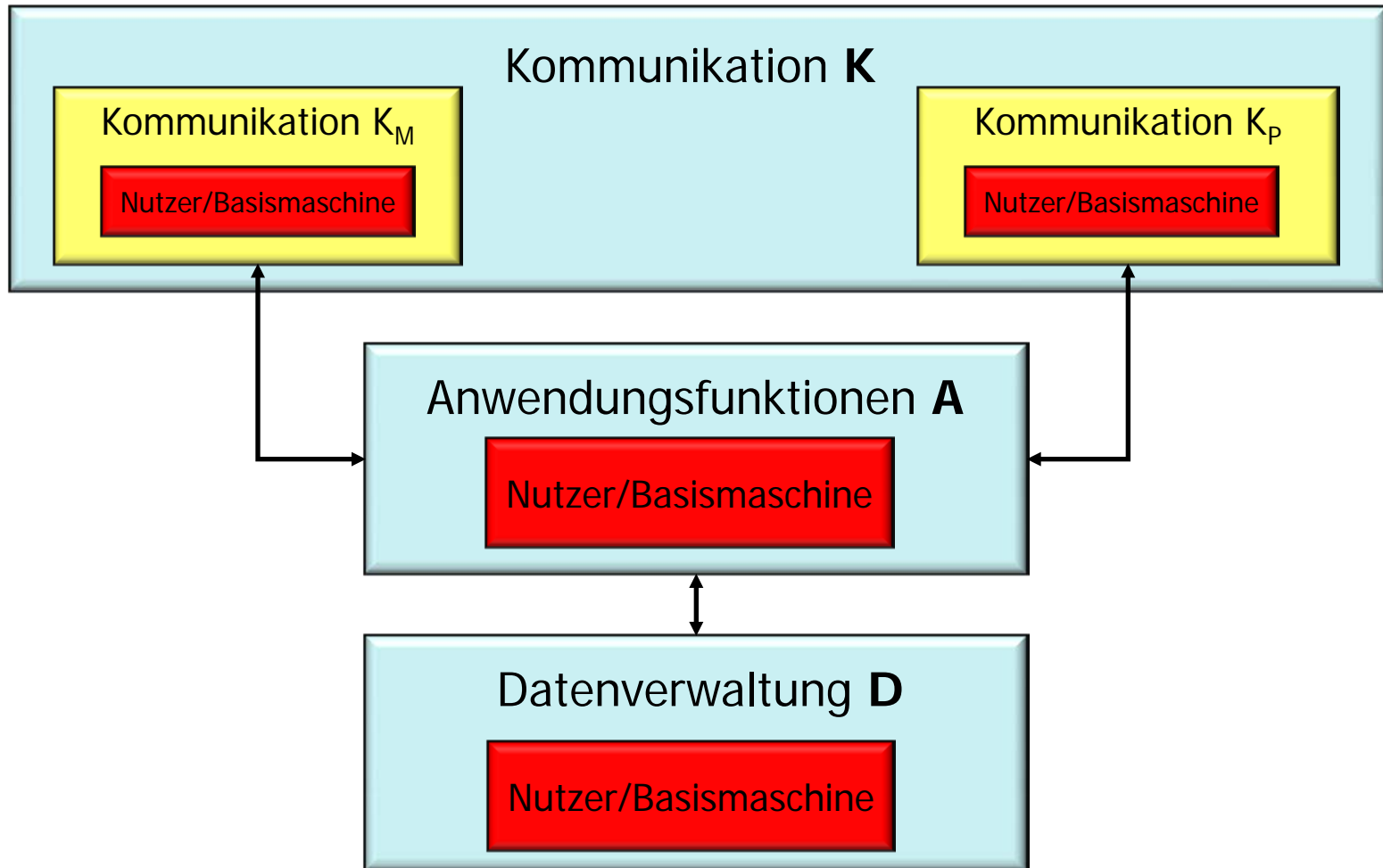




Quelle: Ferstl, O. K.; Sinz, E. J.; Hammel C.; Schlitt M.; Wolf S. (1997)

Textverarbeitungssystem

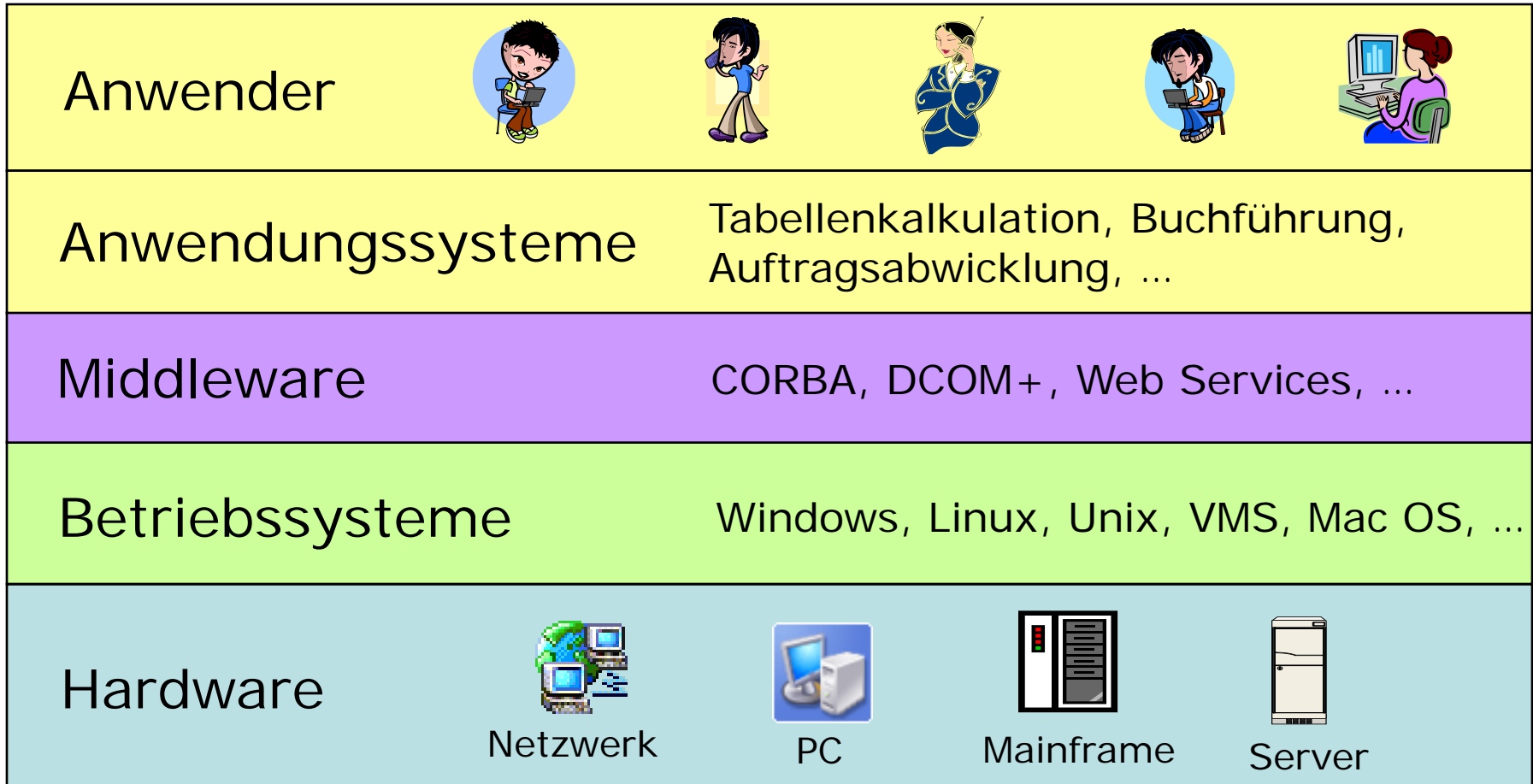




K_P : Kommunikation mit Personen
 K_M : Kommunikation mit Maschinen

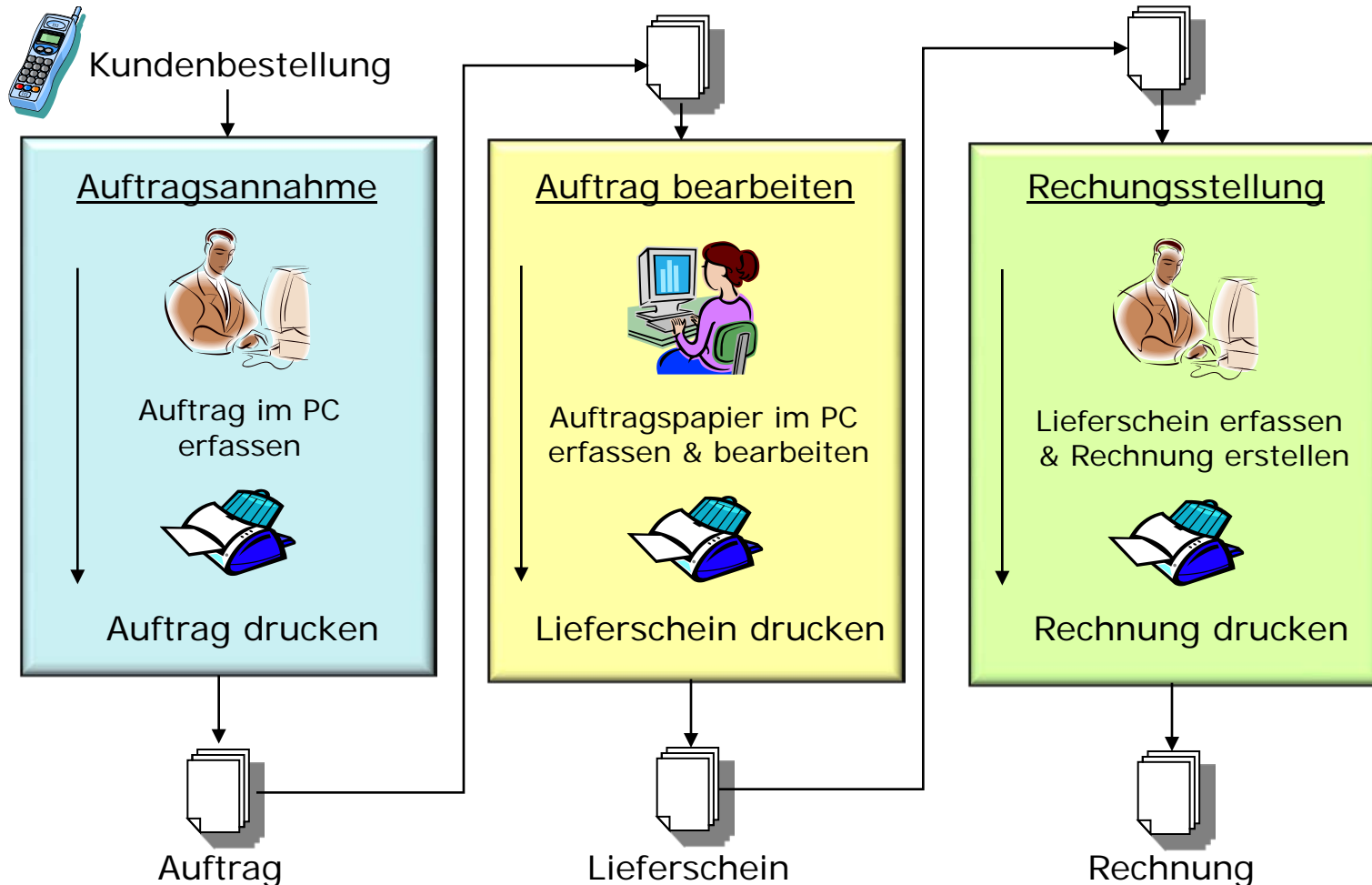
- 1. Modelle für die Architektur von Informationssystemen
- 2. Architekturansätze für Informationssysteme

Typische Komponenten eines Informationssystems



Quelle: Auf Basis von Schwickert, 2003

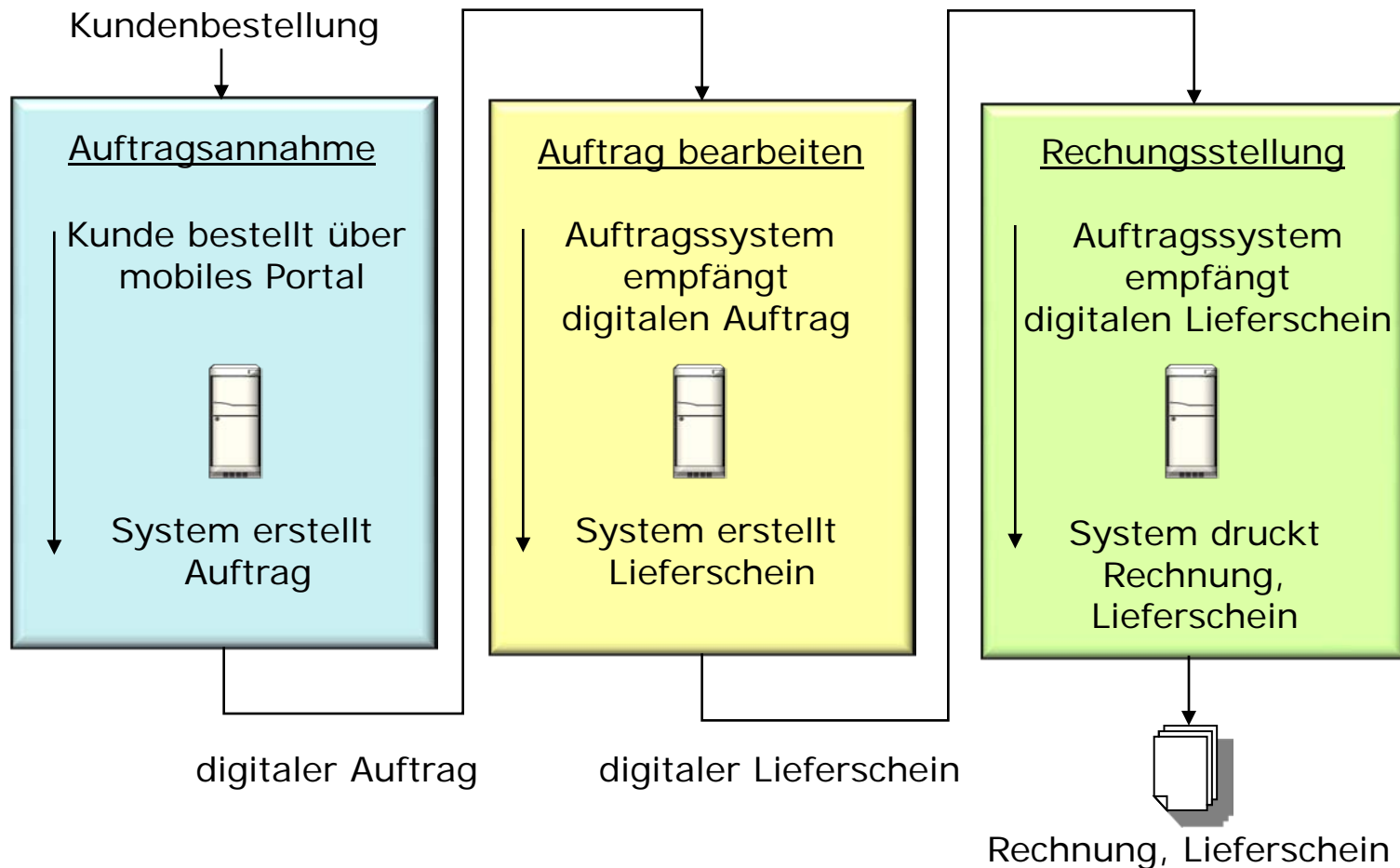
Auftragsabwicklung in einem Unternehmen (Bsp.)



Medienbrüche zwischen den Informationssystemen, d.h.

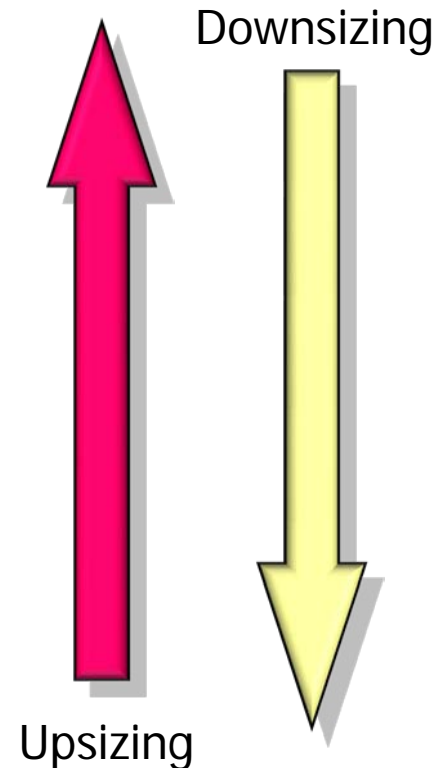
- Lange Bearbeitungszeiten
- Fehleranfällig
- Personalintensiv
- Kostenintensiv
- Unflexibel (z.B. bei Änderung eines Auftrags)
- Schwieriges Controlling, da keine gemeinsame Datenbasis

Auftragsabwicklung in einem Unternehmen (Bsp.)

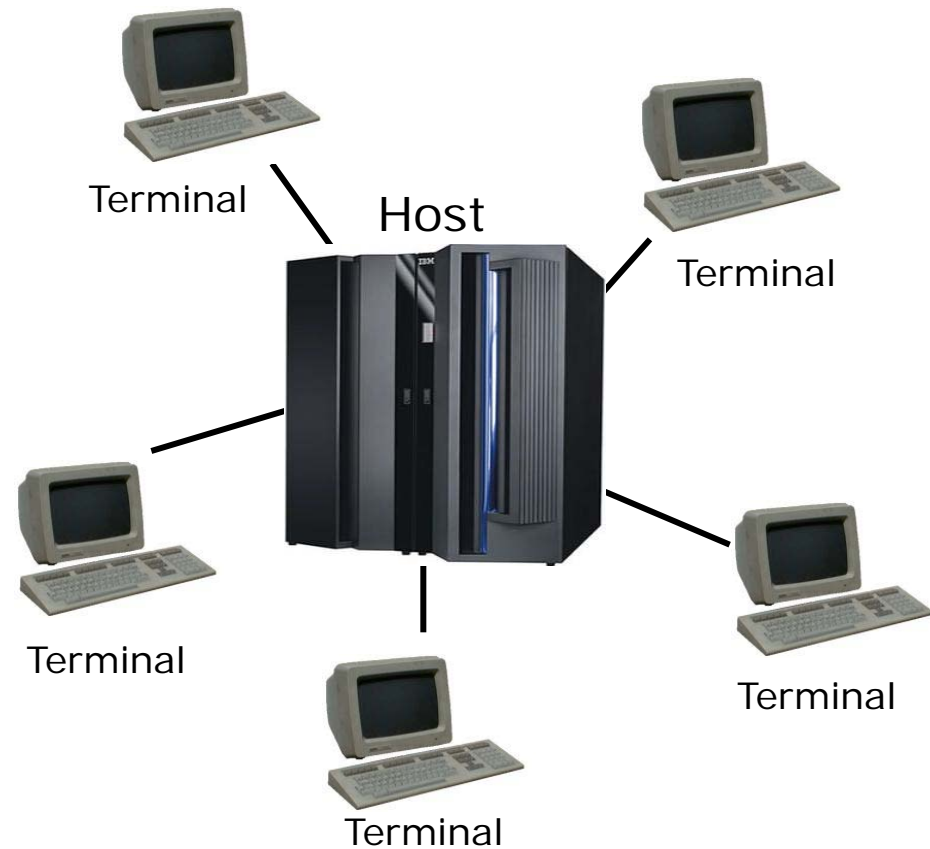


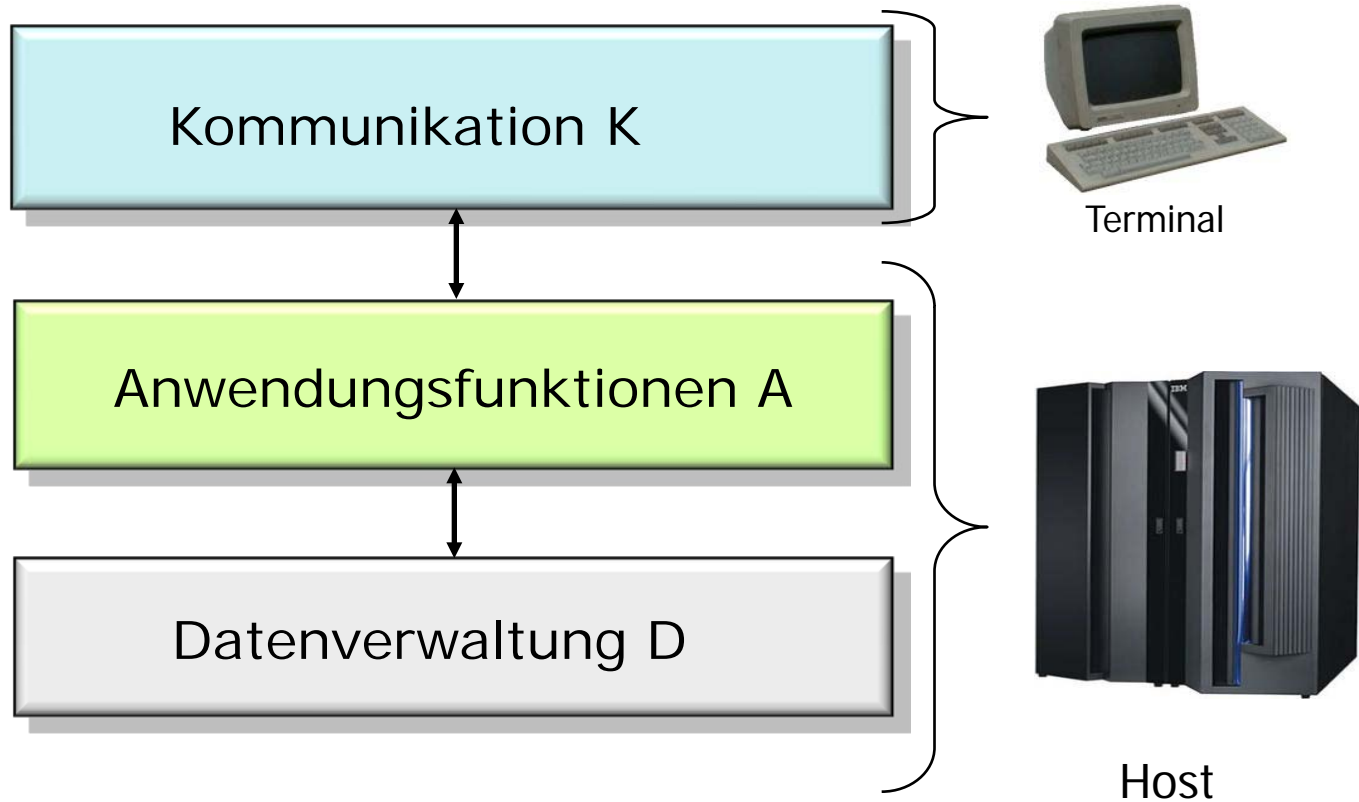
Quelle: Auf Basis von Schwickert, 2003

- **Zentralrechner Konzept**
Zentraler Rechner, auf dem angeschlossene
Terminals arbeiten
- **Ebenen Konzept**
Hierarchisch, über mehrere Ebenen verteilte
Rechner, mit unterschiedlichen
Aufgabenbereichen
- **Client/Server Konzept**
Verwobenes Netz aus Rechnern, welche jeweils
sowohl als Client (Nachfrager von Diensten)
sowie als Server (Anbieter von Diensten)
fungieren können
- **Peer-to-Peer Konzept**
Verwobenes Netz aus gleichberechtigten
Rechnern



- Ein zentraler Host
- Meist „dumme“ Terminals (meist ohne Festplatte)
- Terminals stellen nur die Anwendungsoberfläche dar
- Host führt die Anwendungssysteme aus.
- Zentrale Datenverwaltung durch den Host





- Vorteile
 - Zentrale Datenverwaltung
 - Homogene Anwendungsumgebung
 - Keine Administration der Terminals notwendig
 - Kostengünstige Terminals

- Nachteile
 - Single Point of Failure
 - Starre Struktur
 - Monolithisch
 - Kostenintensive Hosts
 - Problematisch bei großen Datenmengen

Hardware - Hersteller

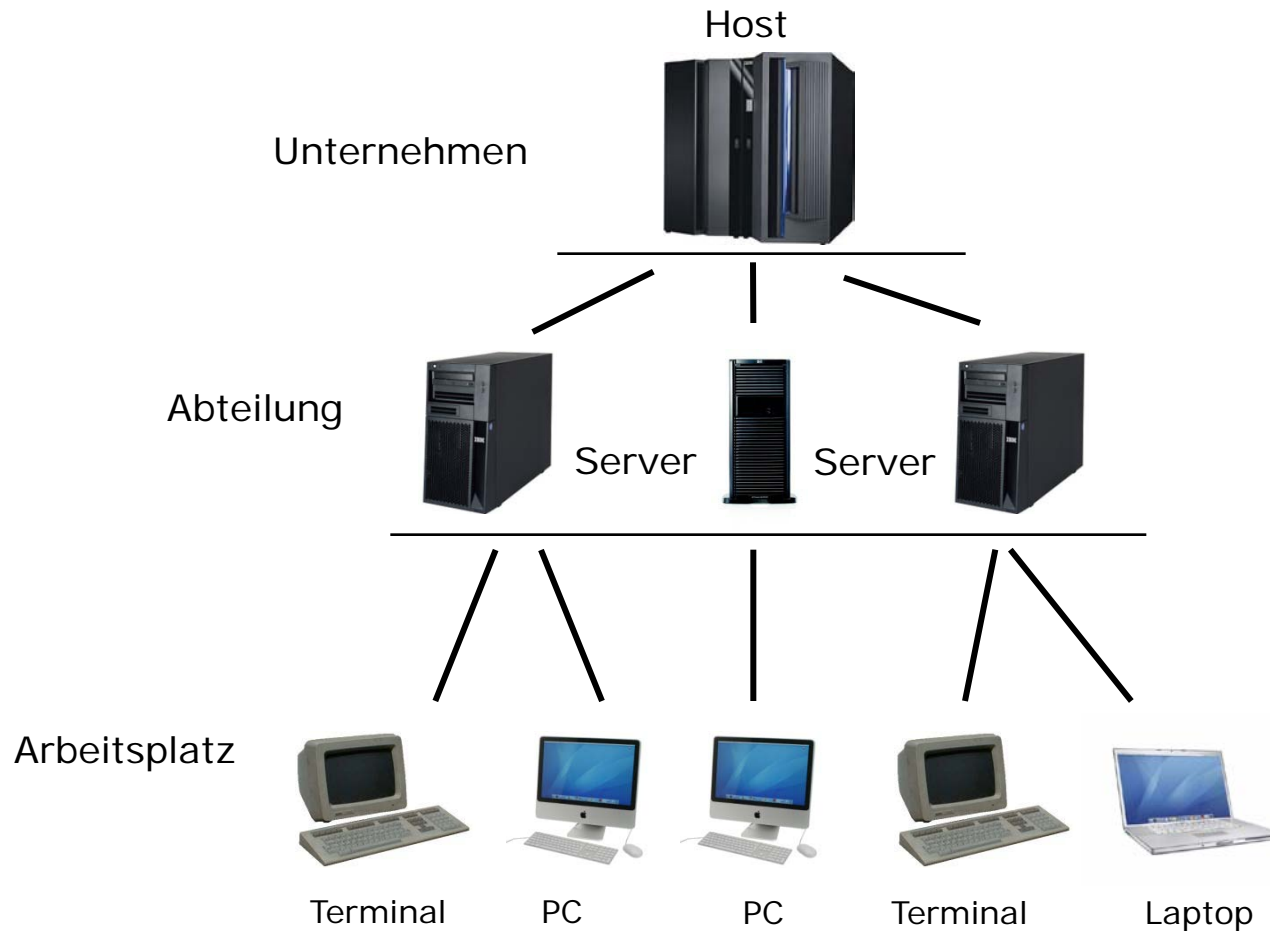


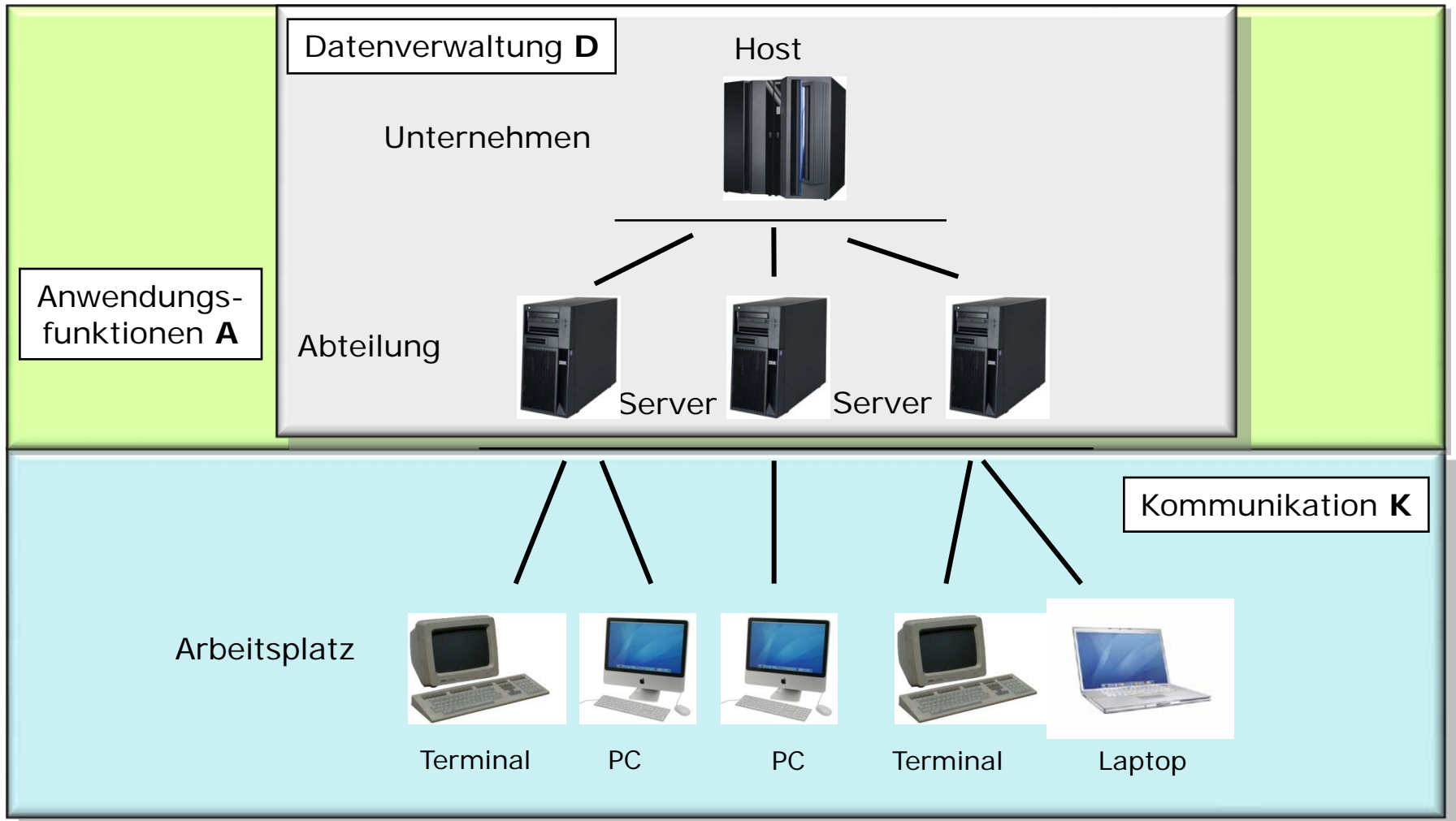
Betriebssysteme

- Unix
- BS 2000
- OS/390
- MVS
- z/OS
- ...



Dezentralisierung nach der Unternehmensstruktur





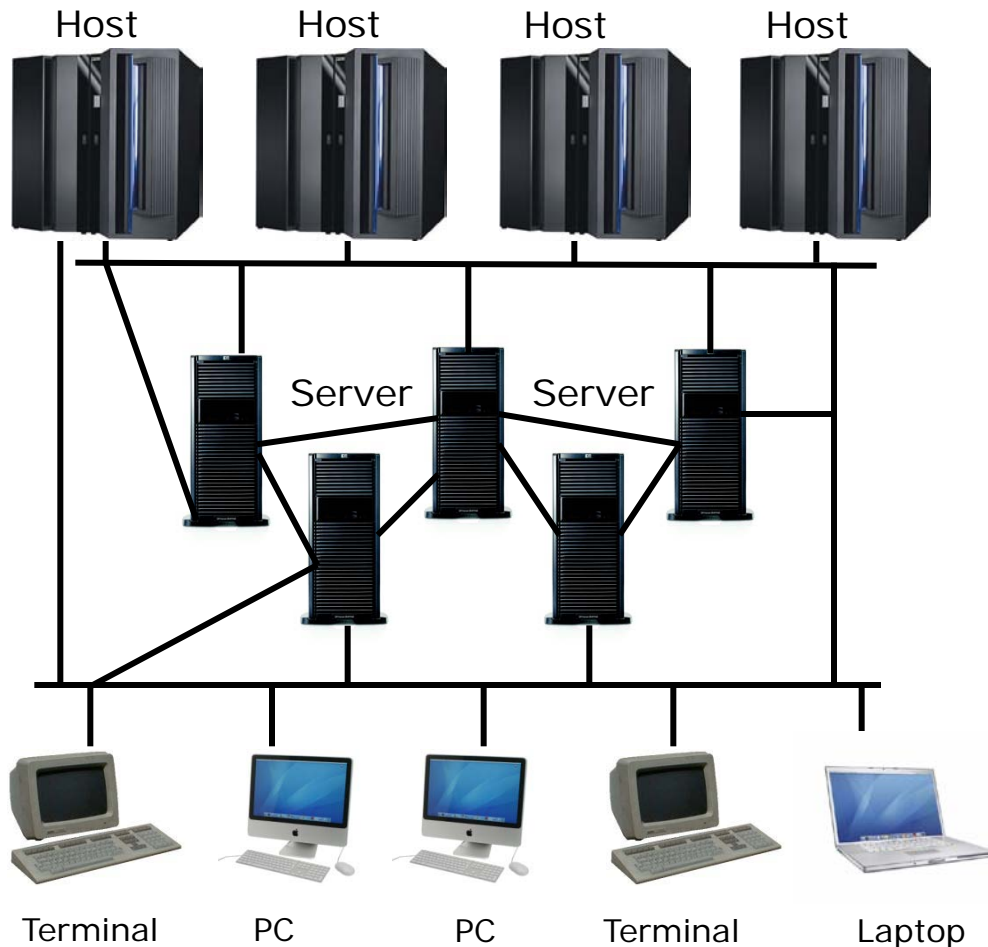
■ Vorteile

- Separierung von Daten und Anwendungsfunktionen nach Zugehörigkeit
- Ausfallsicherheit wird durch Redundanzen erhöht.

■ Nachteile

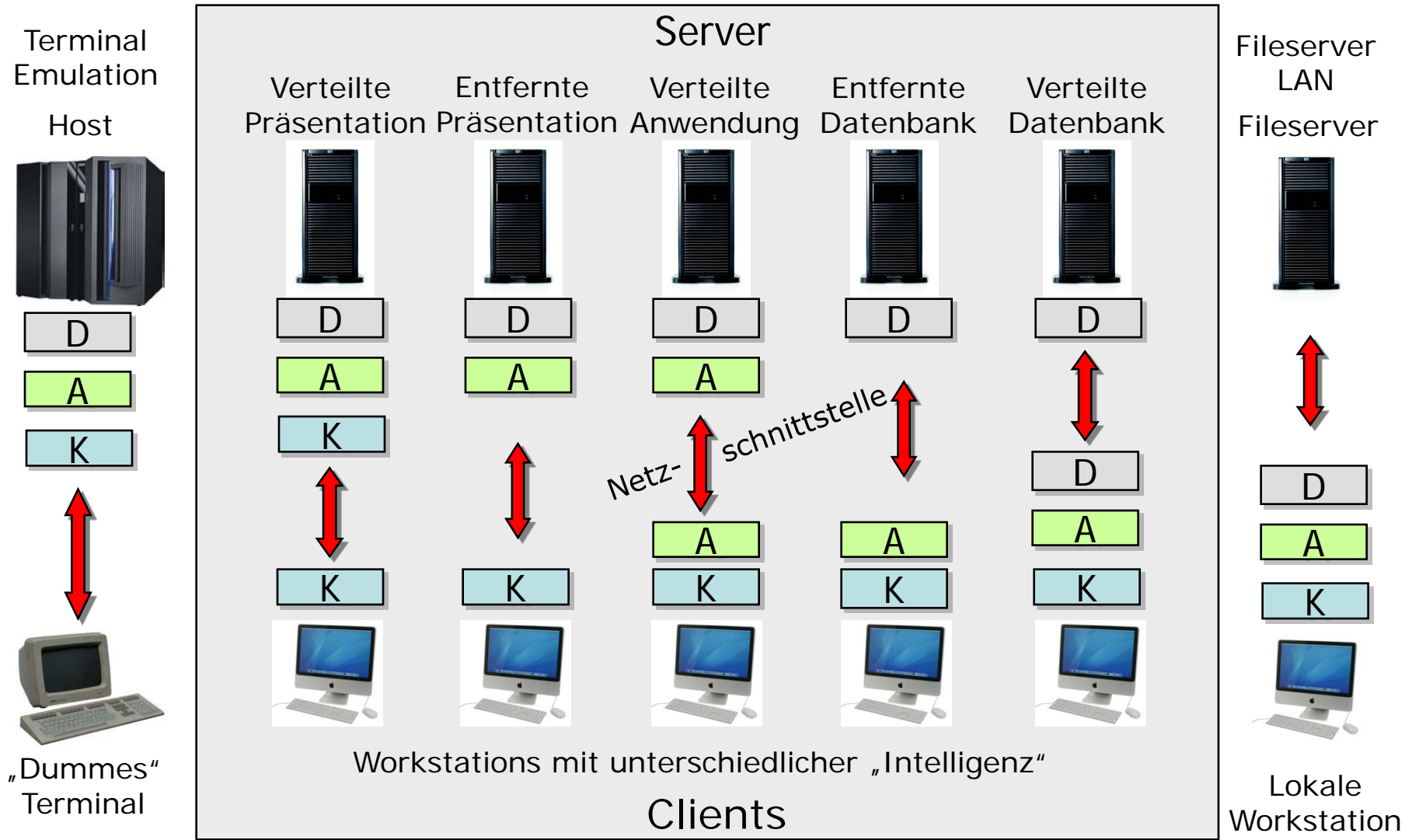
- Übergeordneter Rechner ist immer noch Single Point of Failure.
- Keine direkte Kommunikation zwischen Arbeitsplätzen bzw. Abteilungen möglich

Verwobenes Netz aus Rechnern



- Clients fordern Dienste an.
- Server bieten Dienste an.
- Rechner können Server und Client gleichzeitig sein.
- Ausgeprägte Kommunikation
- Rollenverteilung für (ansonsten autonome) Rechner

Client/Server-Konzept im ADK-Strukturmodell



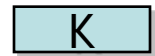
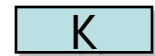
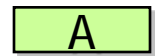
Quelle: Auf Basis von Hennekeuser, 2004

Aufteilung der Kommunikation auf Server und Client

- **Abstrakter Teil der Kommunikation (Server)**
Objekte (z.B. ein Windows-Fenster) werden abstrakt, d.h. ohne konkrete Darstellung und Funktionalität erzeugt.
- **Konkreter Teil der Kommunikation (Client)**
Abstrakte Objekte werden plattformspezifisch bzw. entsprechend der Benutzeroberfläche dargestellt bzw. umgesetzt.
- **Vorteil dieses Ansatzes**
Heterogene Anwendungssysteme können in eine einheitliche Benutzeroberfläche integriert oder auf unterschiedlichen Plattformen verwendet werden.
- **Anwendungsbeispiel: X-Windows**
Eine Benutzeroberfläche kann mit X-Windows auf verschiedenen Plattformen (Windows, Linux, Unix, etc.) dargestellt werden.

Server

Verteilte Präsentation



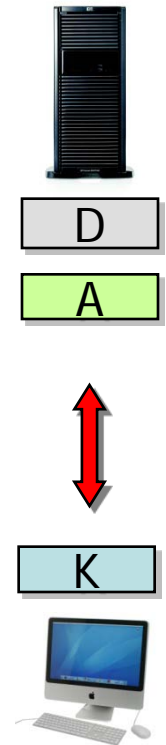
Client

Kommunikation ist auf den Client ausgelagert

- Ausgliederung der Kommunikation auf den Client ist besonders für den Anschluss an Zentralrechner ohne Benutzerschnittstelle geeignet.
- Clients können auf unterschiedlichen Plattformen laufen.
- Benutzeroberflächen können individuell auf den Anwender angepasst werden (z.B. grafisches GUI vs. Shell).
- Client kann kein „dummes“ Terminal sein.
- Beispiel: Citrix Metaframe Access Suite

Server

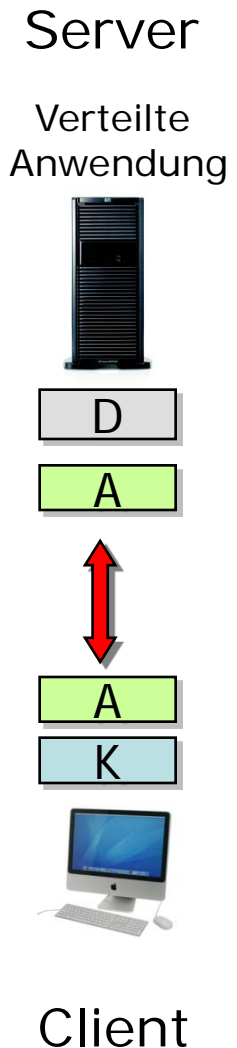
Entfernte Präsentation



Client

Anwendungsfunktionen aufgeteilt: zentral/dezentral

- Zentral genutzte Anwendungsfunktionen werden auf dem Server für eine gemeinsamen Verwendung gehalten.
- Dezentrale Anwendungsfunktionen liegen individuell auf dem Client.
- Zentrale Anwendungsfunktionen werden nur bei Bedarf genutzt.
- Vorteil: Entwicklung und Pflege von Anwendungsfunktionen vereinfacht sich; Komplexität wird reduziert.
- Beispiel: Groupware



Datenverwaltung liegt auf dem Server

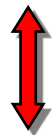
- Klassische Aufteilung für Datenbank-Applikationen
- Mehrere Anwendungssysteme nutzen die gleiche Datenbank/-verwaltung.
- Datenverwaltung kann auch auf mehreren Servern verteilt sein.
- Problem: DB-Abfrage-Standard „SQL“ beinhaltet herstellerabhängig Abweichungen bzw. proprietäre Erweiterungen.
- Klassisches Anwendungsbeispiel: Kundeninformationssystem

Server

Entfernte
Datenbank



D



A

K



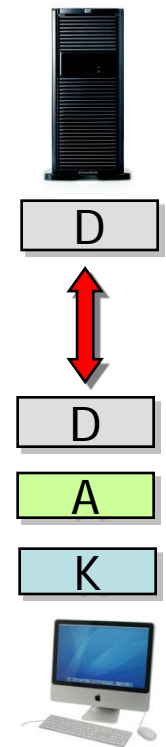
Client

Datenverwaltung verteilt auf Server und Client

- Zwei Ausprägungen der verteilten Datenbank
 - Aufteilung der Daten in zentral (Server) und dezentrale (Client) gehaltene Daten
 - o Organisationsstruktur: Zentrales Adressbuch eines Unternehmens vs. persönliches Adressbuch
 - o Verwendungshäufigkeit: Aktuelle Geschäftsdaten vs. Geschäftsarchiv
 - o Zugriffszeiten: Aktuelle Börsenkurse
 - o ...
 - Aufteilung der Datenverwaltung (DBMS) auf Client und Server
 - o Datenzugriffsfunktionen (häufig genutzt) auf dem Client
 - o Datenbankadministration (weniger genutzt) auf dem Server

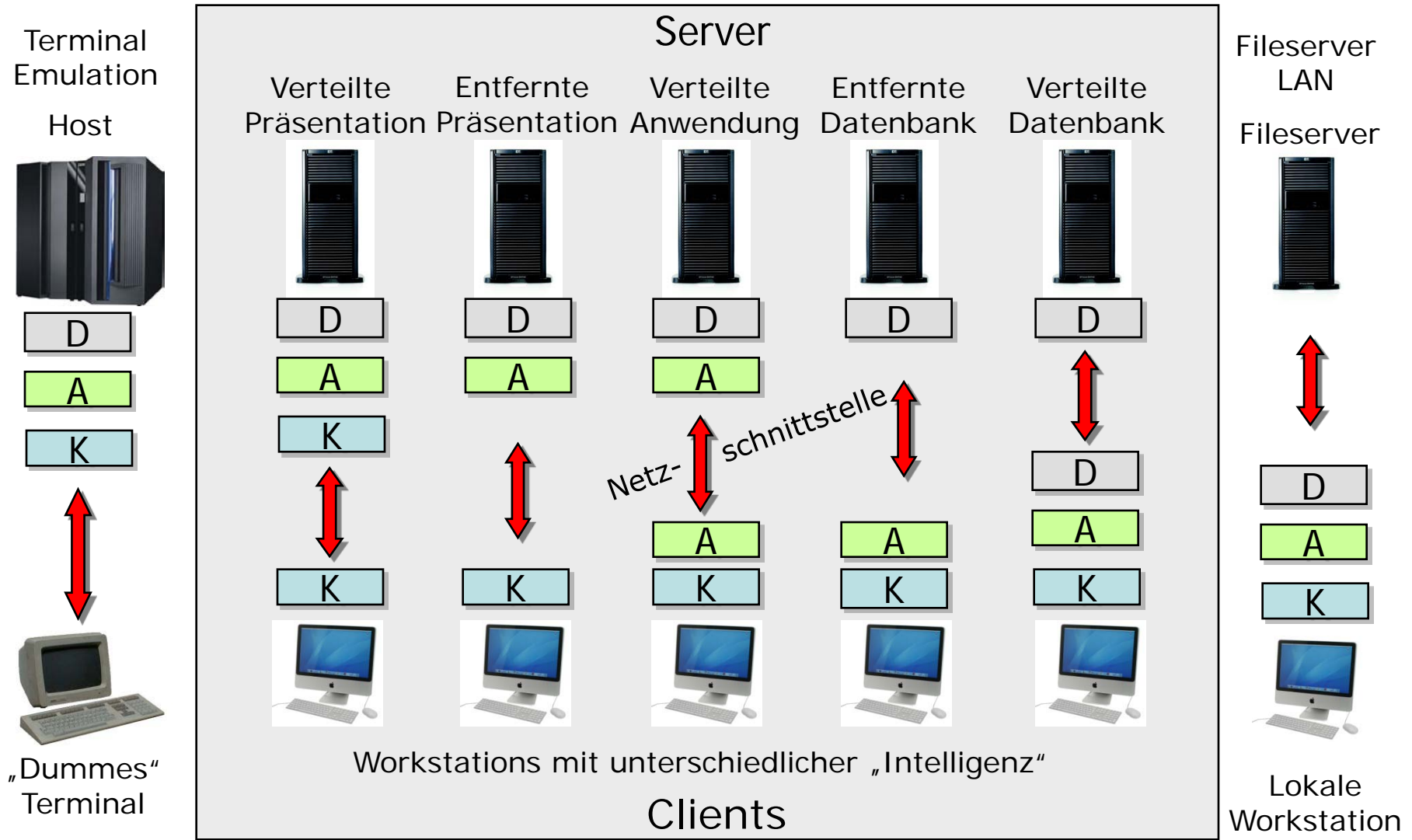
Server

Verteilte
Datenbank



Client

Client/Server-Konzept im ADK-Strukturmodell



Quelle: Auf Basis von Hennekeuser, 2004

■ Vorteile

- Flexibel gestalt- und erweiterbar
- Hohe Kommunikation
- Ausfallsicherheit durch Redundanzen

■ Nachteile

- Hohe Auslastung der Server durch Mehrbenutzerzugriff
- Hoher Planungs- und Koordinationsaufwand
- Breitbandige Netzwerkstrukturen erforderlich
- Hoher administrativer Aufwand

Verwobenes Netz aus gleichberechtigten Rechnern

▪ Eigenschaften

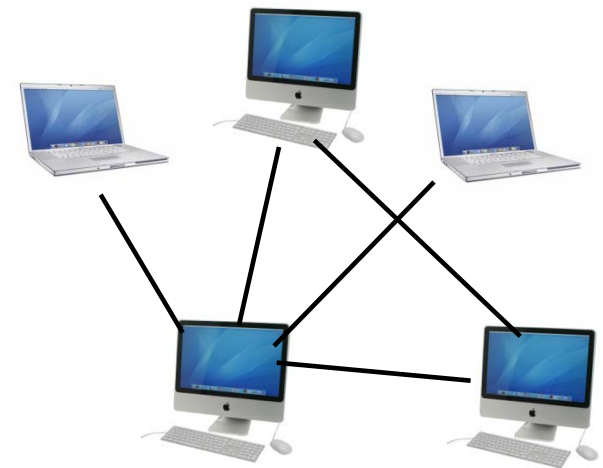
- Keine zentrale Instanz, die Interaktionen steuert.
- Keine zentrale Datenbank
- Peers handeln autonom.
- Jeder Peer kennt nur die Peers, mit denen er gerade kommuniziert.
- Peers, Verbindungen und Informationen innerhalb dieses Konzept sind nicht garantiert.

▪ Vorteil

- Benötigte Ressourcen werden von vielen Parteien erbracht (z.B. Verteilung großer Dateien)

▪ Nachteile

- Hohe Komplexität der Peer-to-Peer Systeme
- Benötigt kritische Masse an Peers
- Sicherheitsaspekte



- Ferstl, O. K.; Sinz, E. J.; Hammel C.; Schlitt M.; Wolf S. (1997) "Applications Objects – fachliche Bausteine für die Entwicklung komponentenbasierter Anwendungssysteme", Bamberger Beiträge zur Wirtschaftsinformatik.
- Ferstl, O. K.; Sinz, E. J. (2001) "Grundlagen der Wirtschaftsinformatik", 4. Auflage, München.
- Hennekeuser J.; Peter G. (2004) "Rechner-Kommunikation für Anwender", Springer Verlag, Berlin.
- Schwickert, A. (2003) "Grundzüge der Wirtschaftsinformatik", Universität Gießen.