23 Architektur verteilter Systeme

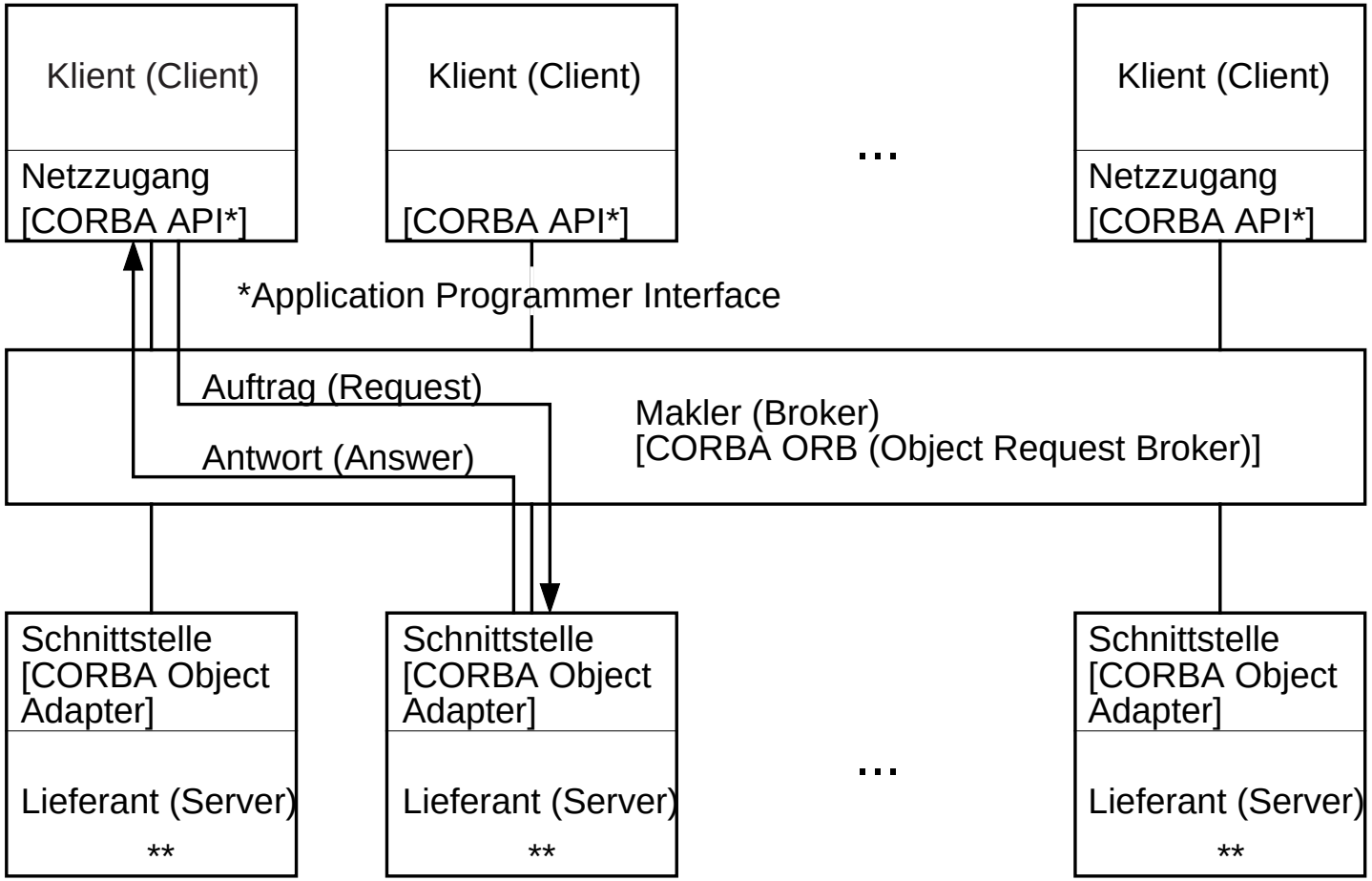
23.1 Grundlagen

- ☆ Warum verteilte Systeme?
  - ❍ Realisierung autonomer, kooperierender Systeme
  - ❍ Anpassung an geographisch verteilte Datenhaltungs- und verarbeitungsbedürfnisse
  - ❍ Leistungssteigerung durch parallele Bearbeitung von Aufgaben
  - ❍ Erhöhung der Zuverlässigkeit eines Systems
- ☆ Entwurfsprobleme bei verteilten Architekturen
  - ❍ Systemtopologie
  - ❍ Aufgabenverteilung
  - ❍ Grad der Kopplung
  - ❍ Datenredundanz

23.2 Verteilte Objekte – Client/Server-Architektur

- Geographisch verteilte Komponenten kommunizieren über Makler miteinander
- Komponenten werden über Schnittstelle angesprochen
- Schnittstelle ist programmiersprachen- und implementierungsunabhängig definiert
- Komponenten sind stark entkoppelt: kennen weder die Implementierung noch die geographische Lokalisierung der Partnerkomponenten.
- Beispiel: CORBA
- Realisiert häufig eine Client/Server-Architektur oder Middleware-Architektur

Client/Server-Architektur unter Verwendung von CORBA



**Typisch Datendienste wie Datenbanksystem(e), Dateisystem(e), Bildarchiv(e),...

23.3 Middleware-Architekturen

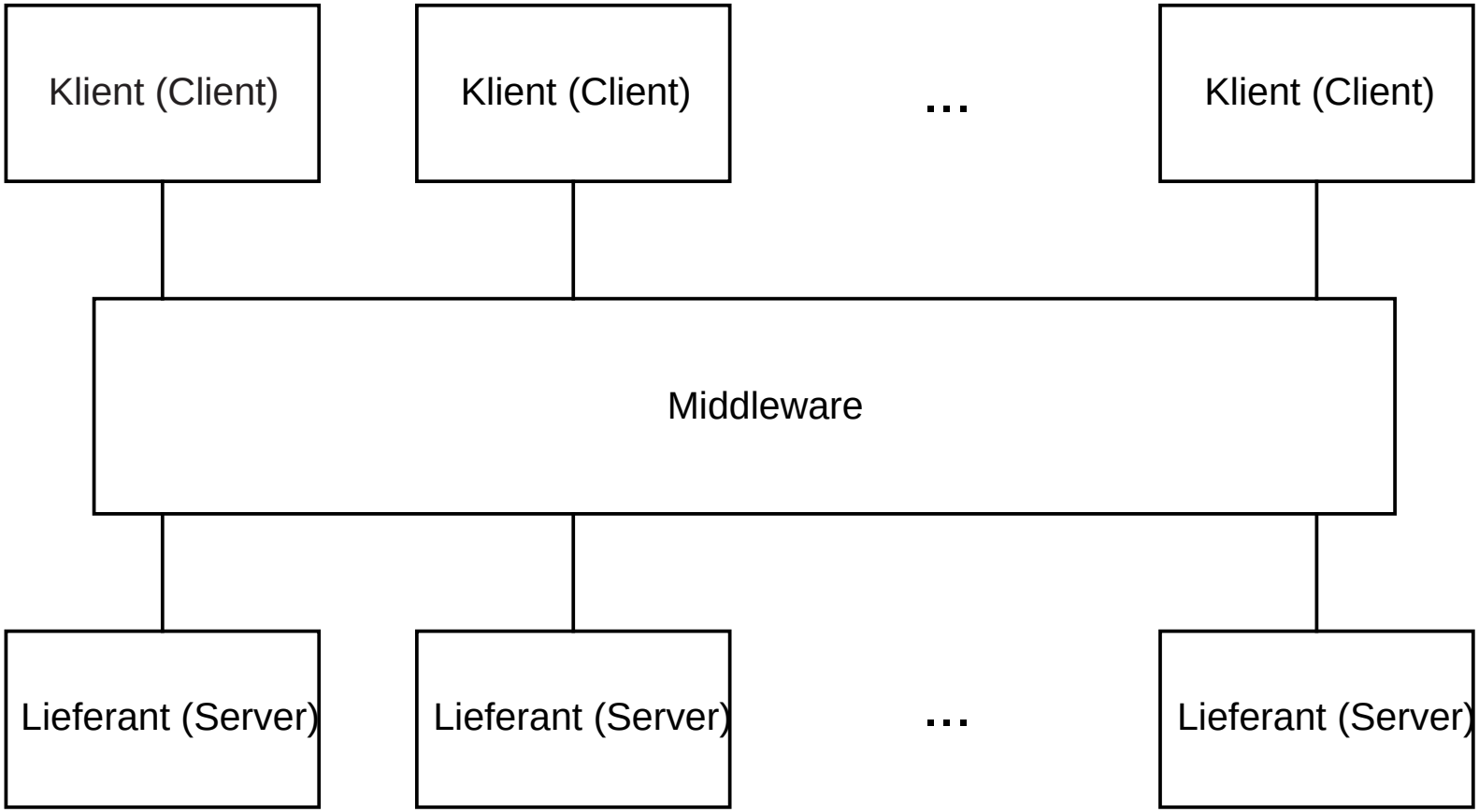
Drei-Schicht-Architekturen:

- ☆ Klienten (Clients)
- ☆ Dienste (Middleware)
- ☆ Lieferanten (Servers)

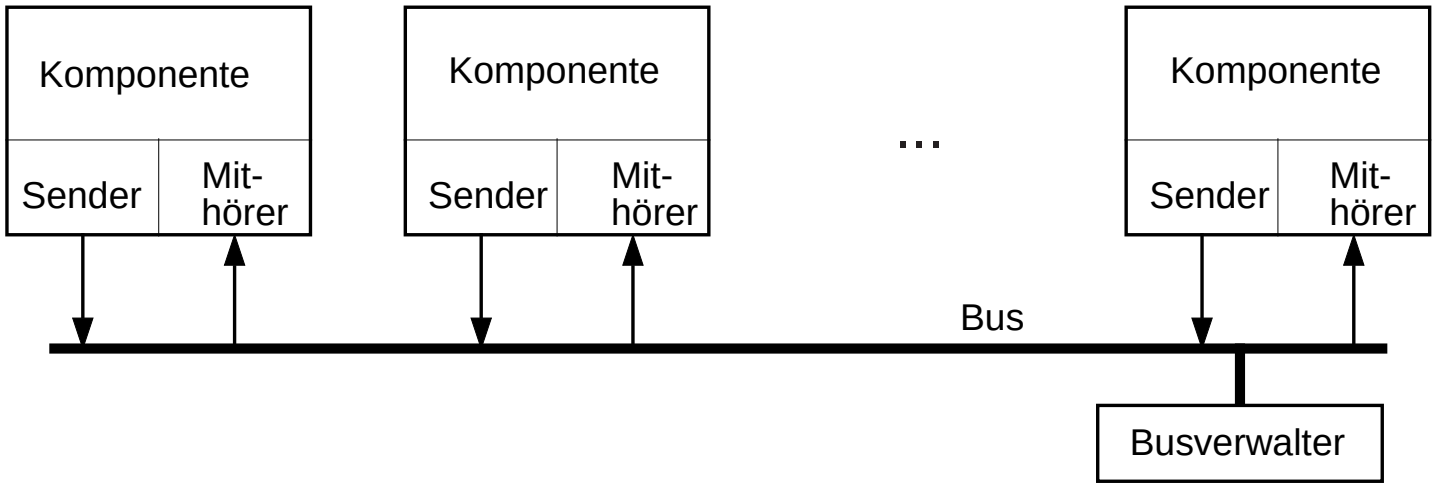
Middleware-Dienste:

- ❍ Kommunikation
- ❍ Schutz und Sicherheit
- ❍ Informationsdienste: Verwaltung, Aufbereitung, Koordination
- ❍ Ablaufsteuerung
- ❍ Präsentation
- ❍ Berechnungen

Prinzip einer Middleware-Architektur



23.4 Bus-Architektur



- Steckbare Komponenten auf Bus
- Globaler Nachrichtenversand über Bus
- Sehr flexibel, im laufenden Betrieb änderbar
- Einsatz u.a. in der industriellen Prozessleittechnik