



Rechnernetze I

SoSe 2025

Roland Wismüller
Universität Siegen
roland.wismueller@uni-siegen.de
Tel.: 0271/740-4050, Büro: H-B 8404

Stand: 1. April 2025



Rechnernetze I

SoSe 2025

8 Datendarstellung



Inhalt

➔ *Marshalling*, Designalternativen

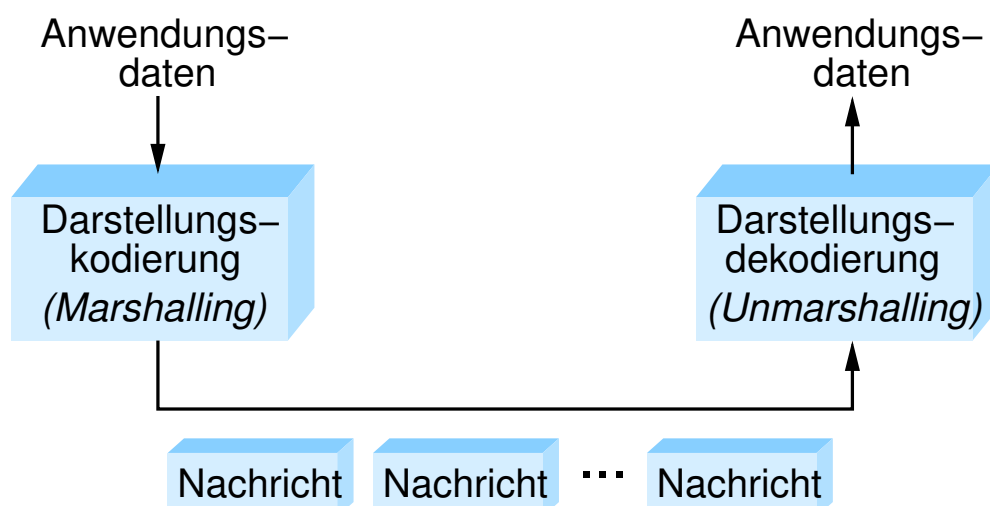
➔ Peterson, Kap. 7.1

8.1 Marshalling



➔ Daten der Anwendungen müssen in eine für die Übertragung geeignete Form umgewandelt werden:

- ➔ **Darstellungsformatierung**
- ➔ ***Marshalling / Unmarshalling***

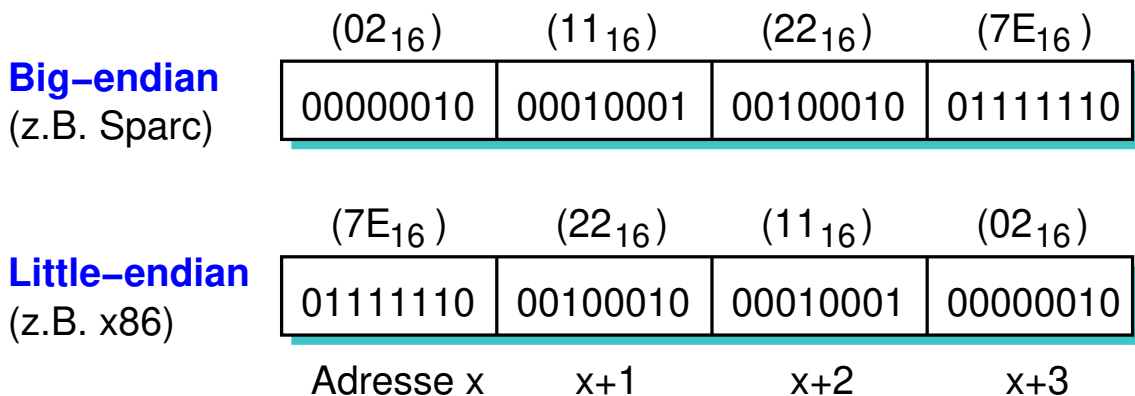


8.1 Marshalling ...



Problem: Heterogenität

- ➔ Rechner haben unterschiedliche Datendarstellung
- ➔ Z.B. einfache 32-Bit Ganzzahl
 - ➔ $34.677.374 = 0211227E_{16}$ (Hexadezimal)



8.1 Marshalling ...



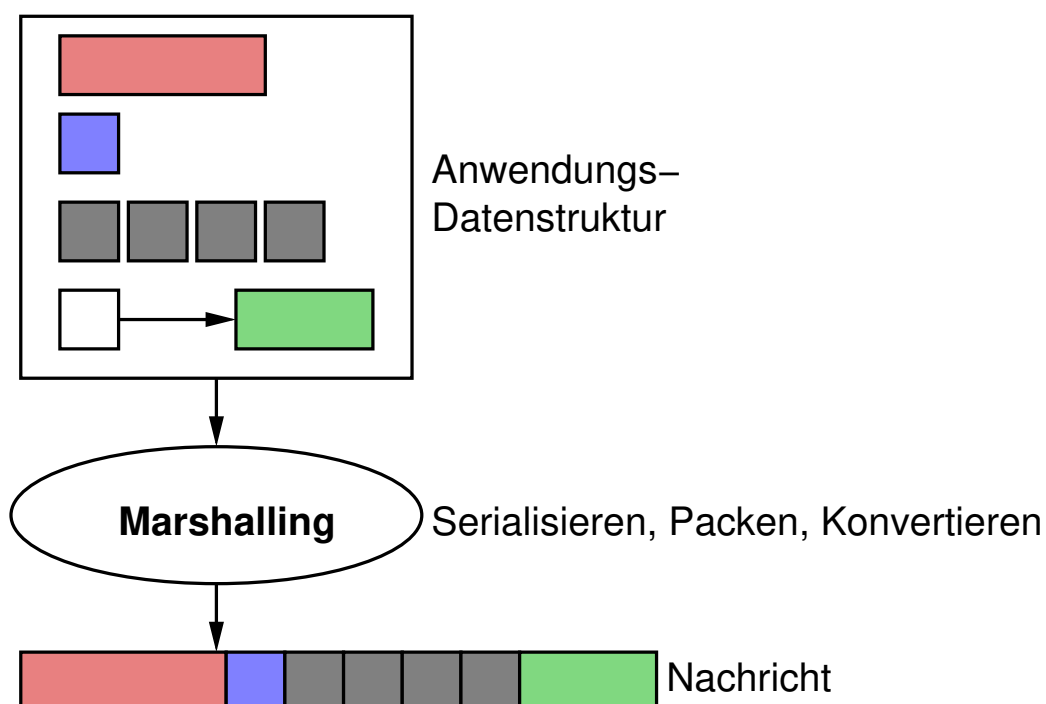
Unterschiede in der Datendarstellung

- ➔ Ganzzahlen:
 - ➔ Größe (16, 32, 64 Bit)
 - ➔ Byte-Reihenfolge: *Big Endian* vs. *Little Endian*
- ➔ Gleitkommazahlen
 - ➔ IEEE 754 vs. proprietäre Formate
 - ➔ Größe (bes. bei *extended precision*: 80, 96, 128 Bit)
 - ➔ Byte-Reihenfolge
- ➔ Datenstrukturen:
 - ➔ *Alignment* der Komponenten im Speicher
 - ➔ z.B. darf eine 16-Bit Zahl an einer ungeraden Adresse beginnen?

Datentypen beim *Marshalling*: drei Ebenen

- ➔ **Basistypen:** int, double, Boolean, char, ...
 - ➔ Formate und Byte-Reihenfolge konvertieren
- ➔ **Flache Datentypen:** Strukturen, Arrays
 - ➔ Füllbytes für *Alignment* entfernen (**packen**) bzw. einfügen (**auspacken**)
- ➔ **Komplexe Datentypen (mit Zeigern):** Listen, Bäume, ...
 - ➔ **Serialisierung** der Datenstruktur notwendig
 - ➔ Zeiger sind Speicheradressen, sie können nicht übertragen werden
 - ➔ Empfänger **deserialisiert** wieder

Marshalling veranschaulicht





Konvertierungsstrategien

➔ Kanonisches Netzwerk-Datenformat

- ➔ Sender konvertiert in Netzwerk-Datenformat
- ➔ Empfänger konvertiert dann in sein Format

➔ *Receiver-Makes-Right*

- ➔ Sender sendet in seinem eigenen Format
 - ➔ serialisiert lediglich
- ➔ Empfänger konvertiert in sein Format, falls nötig

➔ Diskussion:

- ➔ *Receiver-Makes-Right* ist N-mal-N-Lösung
 - ➔ jeder Empfänger muß alle Formate kennen
- ➔ Meist sind die Formate aber identisch (N ist klein)



Datentyp-Kennzeichnung (Tags)

➔ **Tag**: Information, die hilft, die Nachricht zu dekodieren, z.B.:

- ➔ Typ-Tag
- ➔ Längen-Tag (etwa für Arrays)
- ➔ Architektur-Tag (für *Receiver-Makes-Right*)

➔ Beispiel: 32-Bit-Ganzzahl mit Tags:

Typ = int	Länge = 4		Wert = 417892	
--------------	--------------	--	---------------	--

➔ Oft kann man auch ohne Tags arbeiten:

- ➔ Sender und Empfänger wissen, was übertragen wird
- ➔ Darstellungsformatierung dann aber Ende-zu-Ende

- ➔ Daten werden in Rechnern unterschiedlich dargestellt
- ➔ Bei der Übertragung ist eine Anpassung erforderlich
 - ➔ 3 Schritte: Serialisieren, Packen, Konvertieren
- ➔ Realisierungs-Alternativen:
 - ➔ kanonisches Format vs. *Receiver-Makes-Right*
 - ➔ mit / ohne Datentyp-Kennzeichnung

Nächste Lektion:

- ➔ Anwendungsprotokolle