Verteilte Systeme

...für C++ Programmierer

Daten und Interoperabilität

by

Dr. Günter Kolousek

Interoperabilität

- Ausmaß in dem zwei Implementierungen von Systemen nebeneinander existieren und zusammenarbeiten können, indem sie sich auf die Dienste des anderen verlassen, die gemäß einer Spezifikation implementiert sind.
- Aspekte
  - Format der Nachrichten
    - Format der Daten
  - Abfolge (Reihenfolge) der Nachrichten
    - Zeitliche Spezifikationen
  - Spezifikation der Fehlersituationen

  $\rightarrow$ Protokoll

Interoperabilität – 2

- Prozessor, Betriebssystem, Programmiersprache, Netzwerk
- aber auch: Sprachen, Regionen,…
- Lösungsmöglichkeiten (bzgl. Kommunikation)
  - single-canonical format (maschinen)
  - receiver-makes-it-right

Format der Daten

- Textdaten
  - Kodierung und “endianess”
  - Reihenfolge der Zeichen, Format von Datum, Zahlen, Währungen,…
- Binärdaten
  - “endianess”
  - Bitnumbering
  - ganze Zahlen
    - Länge eines `int`, `long`,…
    - signed vs. unsigned
    - Zweierkomplement vs. Einerkomplement
  - Fließkommazahlen
    - IEEE-Formate oder eigene Formate (für Spezialanwendungen)
    - auch hier: Länge eines `double`, `float`
  - Ausrichtung (alignment): z.B. bei Strukturen

In the beginning

- Schreibmaschinen und dann

In the beginning

- Schreibmaschinen und dann Fernschreiber
- ASCII
  - zugeschnitten auf den angelsächsischen Sprachraum!
  - 7 Bits
- “extended ASCII”
  - verschiedene Erweiterungen auf 8 Bits

Probleme bei Textdaten

- Verschiedene Länder — verschiedene Zeichen
  - 8 bit ANSI Code mit Codepages
    - nur ersten 128 Zeichen gleich
    - verschiedene Bedeutung der zweiten Hälfte der verfügbaren Zeichen
  - ISO 8859: 8 Bits $\rightarrow$ 256 Zeichen
    - ISO 8859-1 (latin-1): westeuropäische Zeichen
    - ISO 8859-15 (latin-9): westeuropäische Zeichen $\cup$ {€, ¢,…} $\setminus$ “Unnötiges”
- Länder mit mehr als 256 Zeichen
  - Chinesisch, japanisch
  - multi-byte code pages $\rightarrow$ UTF-8
- Schreib- bzw. Leserichtung
  - z.B.: arabisch (rtl) $\Leftrightarrow$ französisch (ltr)

Probleme bei Textdaten – 2

- Suche
  - z.B. ⅛ vs. 1/8
- Sortierung
  - z.B. Umlaute
- Zahlendarstellungen
  - z.B. 12 756,2 km vs. 12’756.2 km
- Währungen (Symbole oder Abkürzung, Position der Symbole)
  - z.B. € vs. EUR
  - z.B. Schweizer Franken: Fr. vs. SFr. vs. CHF
  - z.B. EUR 42 vs. 42 EUR

Probleme bei Textdaten – 3

- Kalender
  - meist: Gregorianischer Kalender
  - aber auch: Julianischer Kalender, Islamischer Kalender, Jüdischer Kalender, Chinesischer Kalender, Maya Kalender,…
- Datumsformate
  - 24.12.2015 vs. 2015-12-24 ($\rightarrow$ ISO 8601) vs. 12/24/2015
- Zeitformate
  - 14:00 vs. 02:00 p.m., 00:00 vs. 12:00 a.m.
- Zeitzonen
- Telefonnummern, Format von Adressen und Postleitzahlen

Internationalisierung

internationalization, i18n

- Setzen von Maßnahmen, die es ermöglichen, SW in verschiedenen Regionen einzusetzen $\rightarrow$ SW Entwicklung
- technische Maßnahmen
  - Zeichenkodierungen
  - verschiedene Fonts und Textlängen!
  - Links-rechts vs. Rechts-linksschreibung
  - Sortierung, Zahlenformate, Datums- und Zeitformate,…
  - Papiergröße (Letter vs. ISO A4)

Lokalisierung

localization, l10n oder L10n

- Anpassung einer Software an eine spezielle Region
- verwendet die Ergebnisse der Internationalisierung
- Tätigkeiten
  - Übersetzung von Texten (Dialekte, Mehrzahl, Punktierungszeichen (z.B. Anführungszeichen), Fehlermeldungen,…)
  - Tastaturkürzel
  - Titeln, Telefonnummern, Postleitzahlen, Adressen,
  - Anpassungen von Farben, Titeln, Bildern, Filmen
- “Locale”: Konfiguration, die Parameter für eine Region enthält

Unicode Transformation Format

- UTF-8
  - Kodierung mit variabler Länge (1-4 Bytes)
  - Kein Problem mit “endianess”
    - $\rightarrow$ Folge von Bytes
- UTF-16
  - Kodierung mit variabler Länge (2 oder 4 Bytes)
  - Problem mit “endianess”
- UTF-32
  - Kodierung mit fixer Länge (4 Bytes)
  - Vorteil: Zugriff über Zeigerarithmetik auf beliebiges Zeichen
    - aber nicht bei zusammengesetzten Zeichen (d.h. 1 Zeichen = mehrere Codepoints)
    - aber meist werden Zeichen zeichenweise gelesen!
  - Nachteil: Platzbedarf!!

Probleme

- Endianess: Reihenfolge von übertragenen Bytes
- Bitnumbering: Reihenfolge der übertragenen Bits
- Zahlen

Endianess

- Reihenfolge der Bytes
- big-endian
  - erstes Byte enthält signifikante Bits (z.B. Java, PNG, JPEG, MIPS, Sparc, PowerPC, Motorola, Alpha)
  - weitere Bytes (mit steigender Adresse) enthalten die weniger signifikanten Bits
  - “big end first”
    - z.B. 2015-12-24
  - wird als *network byte order* bezeichnet (IP, TCP, UDP,…)
- little-endian
  - letztes Byte enthält signifikante Bits (z.B. GIF, Intel, Alpha)
    - z.B. 24.12.2015

Byte Order Mark (BOM)

- “endianess” $\rightarrow$ Erkennen der Reihenfolge der Bytes
- unsichtbares Leerzeichen der Länge 0 (kein Umbruch)
  - zero width non-breaking space
- hat Codepoint U+FEFF
  - U+FFEF ist reserviert $\rightarrow$ daher kann Reihenfolge der Bytes erkannt werden
    - FEFF für UTF-16
    - 0000FEFF für UTF-32
- fehlt BOM
  - RFC 2781 $\rightarrow$ big-endian
    - aber Intel: little-endian!
  - alternativ kann zwecks Erkenneung nach U+0020 (Space) gesucht werden (kommt oft vor)

UTF-16BE und UTF-16LE

- explizite Kodierung als big-endian oder little-endian
  - BOM wird nicht angegeben
- UTF-16BE `00 43 00 61 00 66 00 65` … “Cafe”
- UTF-16LE `43 00 61 00 66 00 65 00` … “Cafe”

Bit numbering

- Bitnummerierung: Reihenfolge der Bits
  - meist transparent
    - d.h. Programmierer muss sich nicht darum kümmern
- wichtig (z.B.) bei serieller Übertragung
- LSB_0 bit numbering
  - LSB wird Bitposition 0 zugewiesen, z.B.

```
7       0
10010101
```

  - RS-232, Ethernet, USB
- MSB_0 bit numbering
  - MSB wird Bitposition 0 zugewiesen, z.B.

```
0       7
10010101
```

  - I^2C

Zeitzonen

- UTC
  - Coordinated Universal Time
    - Kompromiss zwischen Englisch und Französisch
  - Nachfolger von GMT (an sich das Gleiche)
  - UTC Angabe $\equiv$ Westeuropäische Zeit (WEZ)
    - GMT … Greenwich Mean Time)
- MEZ, CET (UTC+01:00)
  - mitteleuropäische Zeit bzw. central european time
- tz database
  - `Europe/Vienna`
- Sommerzeit (daylight saving time)

ISO 8601

- Datum und Zeitformate
- Jahr
  - 2015
- Monat
  - 2015-12
- Woche
  - 2015-W51, 2015W51 (W01 … 1. Woche mit Donnerstag)
- Datum
  - 2015-12-14, 20151214 (“internationales Datumsformat”)
  - 2015-W51-1, 2015W511 (1 … Montag)
  - 2015-348, 2015348 (ordinale Angabe, von 001 bis 365 bzw. 366)

ISO 8601 – 2

- Zeit
  - lokale Zeit
    - 21:33, 2133
    - 21:33.5 (33.5 Minuten)
    - 21:33:44, 213344
    - 21:33:44.250 oder “21:33:44,25” (44.25 Sekunden)
  - Zeitzone
    - 21:33Z (UTC, zero UTC offset)
    - 21:33+01 (MEZ, CET)

ISO 8601 – 3

- Datum und Zeit
  - 2015-12-14T21:33:44
- Datum, Zeit und Zeitzone
  - 2015-12-14T21:33:44+01 (MEZ)
- Intervalle
  - 2015-12-14T21:33:44/2015-12-24T22:00:00
  - 2015-12-14/P10D (10 Tage),
    2015-12-14T21:33:44/P10DT26M16S
    - D (day), M (month), Y (year), W (week), H (hour), M (minute), S (second)