FOSSLC Workshop: Scriptsprachen

Einführung

Tobias Simon

FOSSLC e.V. & Integrated Communication Systems Group

12. Mai 2011

Was sind Scriptsprachen?

Larry Wall (Perl Entwickler):

„A script is what you give the actors. A program is what you give the audience.“

Eigenschaften:

- Scriptsprachen werden interpretiert
- Vereinigung von Spracheigenschaften zur Erhöhung der Produktivität des Entwicklers
- automatische Speicherverwaltung
- integrierte Funktionalität durch „syntaktischen Zucker“
- optional: Dialogmodus mit interaktivem Interpreter

Wozu eigentlich Scriptsprachen?

Ein (unerfahrener) C/C++ Entwickler könnte behaupten:

„Compilierte Programme sind stets *effizienter* als Scripte!“

Laufzeit-Effizienz:

- Rechenleistung ist heute vergleichsweise billig!
- Laufzeitunterschiede oft vernachlässigbar
- Kombination von Scripten und compilierten Programmen

Effizienz des Entwicklers:

- Personalkosten in IT-Unternehmen sind hoch!
- Empirische Untersuchungen zeigen:
  - C: Produktivitätsfaktor = 1
  - C++/Java: Produktivitätsfaktor = 3
  - Python/Ruby: Produktivitätsfaktor = **6**

Wofür sind Scriptsprachen geeignet?

Prototyping:

- kleine Codefragmente zur Überprüfung von Hypothesen / Machbarkeit
- Einweg-Code der nur 1x verwendet, später evtl. adaptiert wird

Software-Integration:

- Verknüpfung spezialisierter, effizienter Programme (Unix-Prinzip) zu komplexerer Software (z.B. durch Shell-Scripte)
- Kommunikation zwischen Script und compiliertem Programm:
  - Pipes (Stdout, Stdin)
  - Shared Memory
  - Sockets (TCP/IP, Unix Domain Sockets)
- unterstützt Wiederverwendbarkeit
- aber: Abhängigkeiten, „Technologiebindung“!

Wofür sind Scriptsprachen nicht geeignet?

Komplexe Algorithmen:

- Hohe Laufzeit- und Speicherkomplexität besser durch compilierte Programme handhabbar
- Parallelisierbarkeit von Algorithmen auf mehrere Prozessorkerne oft nur umständlich zu realisieren

Hardwarenahe Programmierung:

- Zugriff auf Hardware oft nicht vorgesehen, umständlich
- Echtzeit-Anwendungen schwer realisier- und garantierbar;
  - Garbage-Collector kann den Interpreter anhalten
  - Laufzeit von Datenstrukturzugriffen (Hashtabellen usw) nicht vorhersagbar

Beispiele für Scriptsprachen

Zwei Arten:

- Spezialisierte und Universelle Scriptsprachen

Spezialisierte Scriptsprachen:

- PHP (Hypertext)
- Awk (Textverarbeitung)
- Bash-Script (Programmaufruf, Umgebungsvariablen)

Universelle Scriptsprachen:

- Perl
- Ruby
- ***Python***

Ende der Einführung

Fazit:

- Scriptsprachen sind zurecht weit verbreitet
- Sollten sinnvoll den Anforderungen entsprechend eingesetzt werden:
  - Prototyping
  - Softwareintegration

Es folgt:

Dr. Martin von Löwis (Uni Potsdam) zum Thema: