Software entwickeln unter Linux

Tricks mit C

Wenn Entwickler die volle Übersicht über ihre Software behalten wollen, speichern sie die Versionen der Quelldateien in ihren Binaries. CVS übernimmt diese Aufgabe automatisch – weitgehend. Michael Schilli

Inhalt

96 **Verteilt kompilieren**
Wer oft Software kompiliert, verbringt viel Zeit mit Warten. Verteilte Compiler schaffen Abhilfe, indem sie Jobs auf andere Rechner auslagern.

102 **Feder-Lesen**
Von Java aus Tcl-Code benutzen und von Tcl aus auf Java-Objekte zugreifen. Das Tcl/Java-Projekt entwickelt die passenden Erweiterungen.

108 **Perl-Snapshot**
Mit Glade klicken Programmierer ihre GTK-GUIs einfach zusammen. Ein Netzwerkschnüffler mit grafischer Oberfläche ist damit schnell programmiert.

112 **Coffee-Shop**
PDF-Dokumente in Java erstellen muss nicht umständlich sein: Die Bibliothek I-Text bietet die nötigen Routinen, inklusive Hyperlinks und Verschlüsselung.

Bei komplexer Software ist es gar nicht so einfach herauszufinden, welche Version gerade installiert ist. Zwar zeigen viele Programme mit der Option »-v« ihre Versionsnummer an, doch bei den CVS-Versionen hilft das nicht. Denn Binaries sind oft aus hunderten oder tausenden Source-Dateien zusammengebastelt. Die CVS-Version dieser Dateien in Erfahrung zu bringen ist vor allem für Entwickler wichtig.

Hier hilft ein einfacher Trick: Im Programmcode am Kopf jeder Datei platziert der Programmierer den String:

```
static char *RCSID = "@(#) $Header$";
```

Checkt ein Entwickler nun die 42ste Version mit RCS oder CVS in das Repository ein, expandiert »$Header$« zu etwas wie »$Header: /pfad/myfile.c,v 1.42 2004/09/11 18:50:51 mschilli Exp $« und der Compiler nimmt diesen Wert von jeder Quelldatei in das BSS-Segment des ausführbaren Programms auf.

Aus einem Binary lässt sich dieses Lesezeichen dann mit dem »strings«-Kommando extrahieren:

```
strings myprogram | grep '@(#)'
```

Da die Zeichenfolge »@(#)« sonst wohl kaum in den von »strings« aufgestöberten druckbaren Zeichensequenzen vorkommt, lässt sich genau feststellen, aus welchen Quelldatei-Versionen das Programm besteht, wer die einzelnen Komponenten eingecheckt hat und wann das geschehen ist. Wer will, hält sogar das

Datum und den Zeitpunkt der Kompilierung im endgültigen Binary fest:

```
static char *RCSID = "@(#) $Header$ "
" __DATE__ " " __TIME__;
```

»__DATE__« und »__TIME__« sind Makros, die der Präprozessor durch Datum und Uhrzeit ersetzt, bevor der Compiler die aneinander gereihten Strings zu einem einzigen verbindet. »gcc -E myfile.c« oder »cpp myfile.c« zeigen den Zwischenschritt an. *(mwe/fjl)*

Programmierer-Software

Versionsnummern der aktuellen stabilen Releases am 16.09.2004 und alle Änderungen gegenüber dem Vormonat.

Skriptsprachen
Guile 1.6.4
Object Rexx 2.3.3
Perl 5.8.5
PHP 5.0.1
Python 2.3.4
Ruby 1.8.1
Tcl 8.4.7

Java
Blackdown 1.4.2-rc1
IBM Java2 SDK 1.4.2 GA
Kaffee 1.0.7
Sun SDK 1.4.2_04

Compiler
Binutils 2.15
Clisp 2.33.2
Free Pascal 1.0.10
GCC 3.4.2
GForth 0.6.2
GNU Pascal 2.1
Intel C++ 8.1

Libraries
Coin 2.3.0
FLTK 1.1.4
Glibc 2.3.3
Glut 3.7
GTK+ 2.4.6
Libpng 1.2.7
Libsdl 1.2.7
Libxml 2.6.13
Mesa 6.0.1
OpenSSL 0.9.7d
QT 3.3.3
wxWindows 2.4.2
Zlib 1.2.1

Debugger
Code Medic 2.0.2
DDD 3.3.9
GDB 6.2.1
Insight 5.3
Mpatrol 1.4.8
Valgrind 2.2.0

Build-Tools
Autoconf 2.59
Automake 1.9.1
Libtool 1.5.8
GNU Make 3.80
SWIG 1.3.22

IDEs
Anjuta 1.2.2
Code Crusader 4.0.2
C-Forge 4.0
Eclipse 3.0
Emacs 21.3
KDevelop 3.1.0
Komodo 3.0.1
Kylix 3
Motor 3.3.0
Source-Navigator 5.1.4
VIDE 2.00
VIM 6.3
XEmacs 21.4.15