Übersicht

1. Aufbau und Komponenten einer Entwicklungsumgebung:
   (a) Programmiersprachen
   (b) Editoren
   (c) Testwerkzeuge
   (d) Verwaltungstools
   (e) sonstige Utilities
2. Bibliotheken
3. GNU-Projekt
4. Vergleich mit anderen Umgebungen
5. Programmierkonzepte
6. Anforderungen/Leistung
7. Literatur: Bücher, Zeitschriften, On-Line-Dokumentation

Entwicklungsumgebungen

Notwendige Bestandteile einer Entwicklungsumgebung sind mindestens:

- Programmiersprache (Compiler, Interpreter)
- Editoren
- Testprogramme (Debugger, Profiler)
- Verwaltungsprogramme (Abhängigkeiten, Versionen)
- Hilfsprogramme (Beautifier, Konverter u. v. a. m.)

Programmiersprachen

Unter Linux sind eine ganze Reihe von Programmiersprachen kostenlos und von hoher Qualität frei erhältlich:

- C
- C++
- LISP
- Prolog
- Smalltalk
- FORTRAN
- Pascal
- Modula

Viele Programmiersprachen, die an Universitäten entwickelt werden, sind oft zunächst unter UNIX und damit auch unter Linux verfügbar.

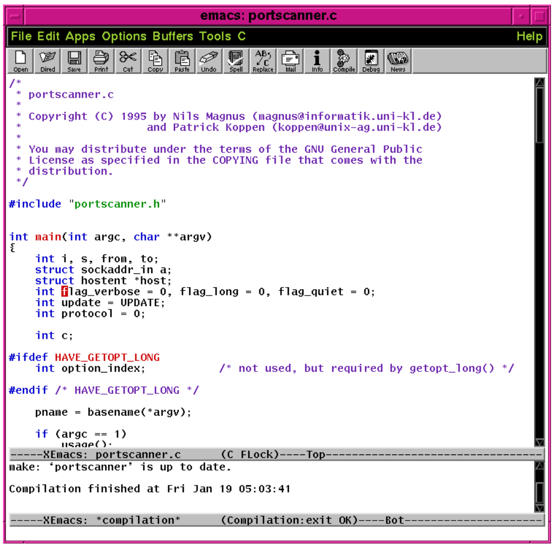
Editoren

Die meiste Zeit während der Programmentwicklung wird vor dem Editor verbracht. Daher ist ein bequem bedienbares und leistungsfähiges Tool notwendig.

- Emacs bzw. XEmacs
  Der Standardeditor unter UNIX bzw. Linux für alle Zwecke. Sollte auf jedem System vorhanden sein.
- aXe
  Ein Editor, der nur mit X-Windows arbeitet. Ähnliche Tatstaturbefehle wie beim Emacs, aber weniger leistungsfähig.
- vi
  Nur für kleinere Änderungen und für "Eingeweihte" brauchbar. Wegen der kleinen Programmgröße startet der Editor schnell, bietet aber auch nur eine kryptische und ungewohnte Bedienung.
- joe
  Ein Wordstar-kompatibler Editor; für alle, die sich nicht von CP/M trennen können.

Emacs

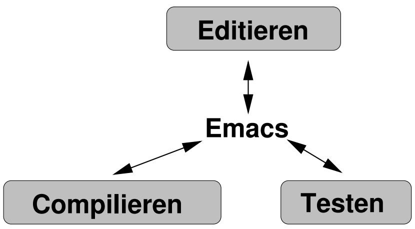
Der Emacs integriert die einzelnen Teile der Entwicklungsumgebung, er steht im Mittelpunkt und ruft die meisten anderen Programme auf:



Übersicht Hilfsprogramme

Programmentwicklung ist kein linearer Prozeß vom Entwurf über das Eingeben des Codes, dem Übersetzung hin zum fertigen Programm. Spätestens bei mittleren Projekten kommt man um den Einsatz von Hilfsmitteln nicht umhin, die eine Verwaltung des Codes vereinfachen und Testläufe und Fehlersuche unterstützen:

- Debugger (`gdb`, `xxgdb`, Emacs-Major-Mode)
- Profiler (`gprof`)
- Abhängigkeitsverwaltung (`make`)
- Versionsverwaltung (RCS, SCCS)
- Beautifier (`indent`)
- Präprozessor, Linker, Assembler

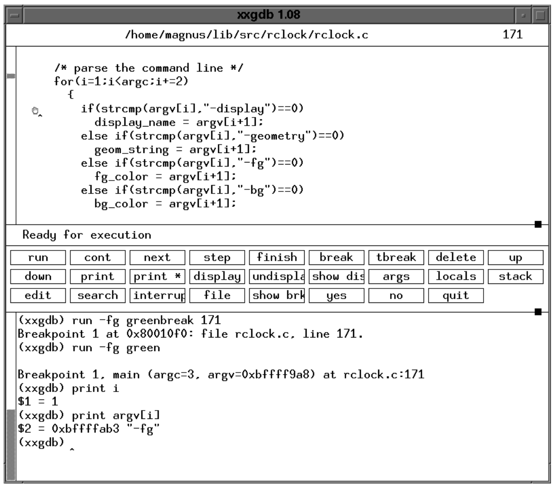


Stärken des Emacs:

- Einrücken von Code,
- sytaktische Überprüfung und Hervorhebung von Konstrukten,
- direktes Einspringen in den Code bei Übersetzungsfehlern,
- Handhabung von mehreren Dateien (Buffer) in mehreren Fenstern (Frames),
- Übernahme von Informationen aus dem Debugger oder anderen Tools,
- sehr weitreichende Steuermöglichkeiten für diejenigen, die sie nutzen möchten.

Debugger

Untersuchung von laufendem Code:



- Breakpoints an Zeilen oder Funktionen
- Anzeige und Veränderung von Variableninhalten

Profiler

Laufzeitanalyse:

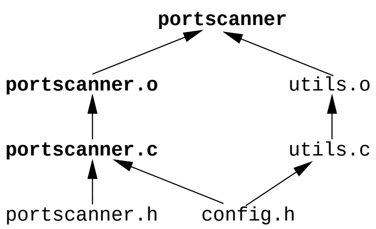
- Untersuchung des Laufzeitverhaltens eines Programmes
- Statistik über Häufigkeit und Dauer von Funktionsaufrufen

```
     Call graph (explanation follows)

granularity: each sample hit covers 4 byte(s) no time propagated

index % time    self  children    called     name
                0.00    0.00       1/7           read_db [4]
                0.00    0.00       6/7           learn_new [2]
[1]      0.0    0.00    0.00       7         new_node [1]
-----------------------------------------------
                0.00    0.00       4/4           work [5]
[2]      0.0    0.00    0.00       4         learn_new [2]
                0.00    0.00       6/7           new_node [1]
-----------------------------------------------
                0.00    0.00       1/1           _start [16]
[3]      0.0    0.00    0.00       1         main [3]
                0.00    0.00       1/1           read_db [4]
                0.00    0.00       1/1           work [5]
-----------------------------------------------
                0.00    0.00       1/1           main [3]
[4]      0.0    0.00    0.00       1         read_db [4]
                0.00    0.00       1/7           new_node [1]
-----------------------------------------------
                0.00    0.00       1/1           main [3]
[5]      0.0    0.00    0.00       1         work [5]
                0.00    0.00       4/4           learn_new [2]
-----------------------------------------------
```


Abhängigkeitsverwaltung

Mit dem Tool `make` kann man Abhängigkeiten zwischen verschiedenen Quelltextdateien eines Projektes angeben.

- Mit nur einem Befehl kann ein ganzes Projekt "gebaut" werden
- Es werden nur die notwendigen Übersetzungs- und Bindeschritte ausgeführt.



Makefiles sind eine "Bauanleitung" für ein Projekt und können auch Regeln zum Aufräumen nicht länger benötigter Zwischendateien oder zur Installtion von Programmpaketen verwendet werden.

Versionsverwaltung

Eine Versionsverwaltung führt Buch über durchgeführte Änderungen im Lebenszyklus eines Softwareprojektes.

- Verwaltung von Versionsnummern
- Speicherung von Kommentaren zu Versionen/Releases
- Wiederherstellung älterer Versionen

Gebräuchlich sind die Pakete SCCS (Source Code Control System) und RCS (Revision Control System) des GNU-Projektes, die beide eine ähnliche Funktionalität haben. Bei Linux ist zumeist RCS vorhanden.

Sonstige Hilfsmittel

Es gibt eine ganze Reihe von Konvertierungsprogrammen, die Quelltexte von einer Form in eine andere bringen:

- Wandlung von K&R-Style nach ANSI-C und umgekehrt (`(un)protoize`)
- Einheitliche Einrückung von Quelltexten (Beautifier, `indent`)

Von vielen weiteren Programmen merkt man normalerweise wenig, da sie im "Hintergrund" arbeiten:

- Der Präprozessor löst Makros auf und erlaubt eine bedingte Übersetzung,
- Assembler übersetzt das Ergebnis des Compilers in ausführbaren Maschinencode,
- Linker fügt übersetzte Programmteile zusammen und erzeugt das ausführbare Binary.

Bibliotheken

Zu einer ganzen Reihe von sich häufig wiederholenden Aufgaben existieren bereits Bibliotheken, die vielfältige Funktionalität zur Verfügung stellen:

- Mathematische Funktionen, höhere Genauigkeit, Statistik
- grundlegende Grafikroutinen(X-Windows, SVGA-Lib)
- Widgetsammlungen (Motif, XForms)
- Grafikformate (JPEG, PNG, pbmtools)
- Support für C++, FORTRAN und Pascal
- Debugging, Aufspüren von Speicherleaks (libchecker, electric fence)

GNU-Projekt
GNU's not Unix

Viele Standardkomponenten eines Linuxsystems sind dem GNU-Projekt entnommen:

- GPL (Gnu Public License) und eine spezielle Lizenz für Bibliotheken garantieren die freie Benutzbarkeit der Programme, Tools und fertigen Programme
- GNU-Tools sind für fast jeden Rechner erhältlich → sehr gute Portabilität
- hält sich an Standards (ANSI-C, ...), ist strikt POSIX-konform, vermeidet Restriktionen (Länge von Bezeichnern etc.)
- hat auf vielen Maschinen eine gute Performance

Ziel des GNU-Projektes ist es, der Allgemeinheit qualitativ hochwertige Programme frei zur Verfügung zu stellen.

Vorteile gegenüber DOS

Linux bietet einige Vorteile gegenüber der Programmentwicklung unter DOS/Windows, wo Anwendungsprogramme zumeist noch die "Nase vorn" haben:

- Professionellere Umgebung,
- Multitaskingfähigkeit,
- Absturzsicherheit durch Speicherschutzmechanismen von UNIX,
- Mächtigere Konzepte verfügbar (Prozesse, Signale, Sockets, ...),
- Bessere Portabilität (z. B. auch Crossentwicklung für DOS),
- Billiger/besser durch freie Software, keine Lizenzen,
- auch für Linux sind integrierte Pakete (`xwpe`) verfügbar.

Mächtige Programmierkonzepte

Mit UNIX sind eine ganze Reihe interessanter und mächtiger Programmierkonzepte verfügbar, die im wesentlichen den "State-of-the-Art" der Programmierung ausmachen:

- Prozesse
  Einfache Abstraktion von mehreren "Benutzern", Realisierung von mehreren Aufgaben "gleichzeitig" (Multitaskting).
- Virtueller Speicher
  Jeder Prozeß hat prinzipiell 4GB Hauptspeicher zur Verfügung — keine Restriktionen mehr wegen "Small Models" o. ä.
- Interprozeßkommunikation
  Mit Signalen und Pipes hat man die Möglichkeit, mit anderen Prozessen des selben Rechners zu kommunizieren.
- Netzwerkzugriff
  Abstraktion vom einzelnen Rechner, transparente Verteilung auf mehrere Rechner unabhängig von deren Aufstellungsort.
- Benutzeroberflächen
  Mit X-Windows steht ein verteiltes und netzwerktransparentes Grafiksystem zur Verfügung.

Hardwareanforderungen/Leistung

- Die Programmentwicklung ist eine relativ genügsame Anwendung.
- Ein 486 DX2/66 mit 16 MB Arbeitsspeicher ist für gelegentliche Aufgaben oder die Anfertigung von Übungsaufgaben schon ausreichend.
- Eine Aufstockung des Hauptspeichers zahlt sich nach wie vor eher aus als bei der Prozessorleistung, zumal die Preise für RAM stark gefallen sind.
- Durch die Unterstützung aller Intelplattformen bis zum PPro auf SMP-Boards kann (mit bspw. einem 17-Zoll Monitor und 64 MB RAM) kaum eine andere Workstation auch im obeneren Leistungsspektrum ein solches Preis-/Leistungsverhältnis bieten.

Literatur

Bücher:

Linux: Installation, Konfiguration, Anwendung: Michael Kofler, Addison-Wesley, incl. CD-ROM, DM 89,90, Inf 468/269
Einsteigerbuch, das auch auf diverse Entwicklungswerkzeuge eingeht

Linux Kernel Programmierung: Michael Beck, Harald Böhme u. a. (3. Auflage), Addison-Wesley, incl. CD-ROM, DM 79,90, Inf 468/270
Für Leute, die tief in die Innereien von Linux einsteigen möchten. Praxis eines Betriebssystems

Learning GNU Emacs: ???, O'Reilly Nutshell Series, ca. DM 79,–
Eine Einführung in die Konzepte des UNIX-Editors

UNIX Network Programming: Richard Stevens, Prentice Hall, ca. DM 90,–
Das Buch über Netzwerkprogrammierung

gcc, libg Documentation GNU-Project, als Info-Files installiert, gebundene Versionen beim GNU-Projekt erhältlich
Eine sehr ausführliche Beschreibung des Compilers und der Implementation der Standard-C-Library

Zeitschriften:

iX Multiuser Multitasking Magazin: Heise Verlag, DM 7,50 pro Heft
Hier erscheinen häufig Reports auch zu Linux-Themen

Linux Magazin: ARTICON GmbH, DM 9,– pro Heft
Erste und bislang einzige deutschsprachige Linux-Zeitschrift; noch in der Entstehungsphase, mit einigen guten Ansätzen

Linux Journal: erscheint in den USA, nur über bestimmte Buchhandlungen oder Direktversand beziehbar.

Online-Dokumentation:

- Bei jeder Linux-Distribution ist eine Menge an Dokumentation beigefügt(HOWTOs, FAQs, Manpages, Info-Files).
- Auf vielen Web-Servern findet sich ein Link auf die Homepage des LDP (Linux Documentation Projekt):
  z. B. `http://www.unix-ag.uni-kl.de/~linux/`

Ausblick