Entwicklungsmethoden

Prof. Dr. Josef M. Joller
jjoller@hsr.ch

TOOLS

Übersicht

- Stufenweise Verfeinerung
- Kosten / Nutzen analyse
- Software Metriken
- CASE
- Taxonomie von CASE
- Umfang von CASE
- Software Versionen
- Konfigurationsmanagement
- Build Werkzeuge

Stufenweise Verfeinerung

Die Stufenweise Verfeinerung ist ein grundlegendes Prinzip, welches in vielen Techniken angwendet wird.

- Grundschema: verschieben Sie alle Entscheide über Details; konzentrieren Sie sich zuerst auf die wichtigsten Fragen

Miller's Gesetz (1956)

- Ein Mensch kann sich in der Regel af 5+/- 2 Dinge gleichzeitig konzentrieren.

Warum stufenweise Verfeinerung?

Dieses Prinzip wird

- in jeder Phase eingesetzt
- für jedwelche Darstellung

Warum stufenweise Verfeinerung

- der Software Engineer kann sich nur auf 5+/-2 Dinge gleichzeitig konzentrieren

Warnung

- Miller's Gesetz gilt universell, nicht nur im Bereich der Software Entwicklung

Kosten / Nutzen Analysen

Vergleich der IST (laufende) Kosten mit der neuen Lösung (Investition plus laufende Kosten)

- Kostenschätzung
- Nutzenabschätzung
- führen Sie alle zu berücksichtigenden Faktoren auf

Quelle

- Boehm : Software Engineering Economics
- eigene Projektdaten

CASE (Computer-Aided Software Engineering)

Einsatz von CASE (Computer Aided Software Engineering

- CASE Tools stehen für jegliche Phasen zur Verfügung

Viele Tools verfügen über grafische Interfaces und Darstellungen

- Datenfluss-Diagramme
- Entity-Relationship Diagramme
- Module-Verbindungs Diagramme
- Petri Netze
- strukturelle Darstellungen

Software Metriken

Damit Fehler möglichst früh erkannt werden, muss man Messungen durchführen, um Abweischungen feststellen zu können.

Beispiele:

- LOC pro Monat
- Anzahl Fehler pro 1000 Lines of Code (LOC)
- Anzahl Bildschirme
- Anzahl Berichte
- Anzahl Dateien
- …

Unterschiedliche Metriktypen

Produktmetrik

- Beispiel:
  - Grösse des Produktes
  - Zuverlässigkeit des Produktes

Prozessmetriken

- Beispiel:
  - Effizienz der Fehlerfindung im Verlaufe der SW Entwicklung

spezifische Metriken für bestimmte Phasen

- Beispiel:
  - Anzahl Fehler pro Reviewstunde

Die fünf grundlegenden Metriken

Grösse

- Anzahl Programmzeilen

Kosten

- in monetären Einheiten

Dauer

- in Monaten

Aufwand

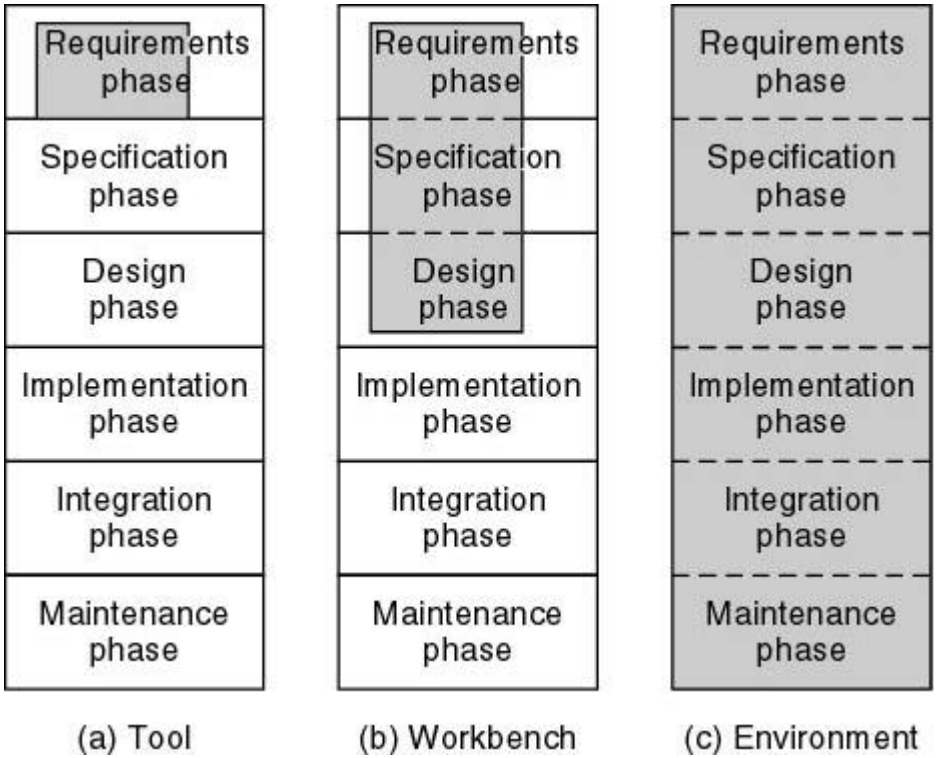
- in Personenmonaten

Qualität

- Anzahl gefundener Fehler

Taxonomie von CASE

UpperCASE versus lowerCASE



Tool versus workbench versus environment

Graphische Tools

Zusatzfunktionen

- Data Dictionary
- Screen und Report Generatoren
- Consistency Checker

Online Dokumentation

- Manuals
- Updating

Wesentliche Online Dokumentation

- Help Information
- Programmier-Standards
- Manuals

Tools für den Programierer

Programmier Tools

- Text Editoren,
- Debugger
- Interface Checker

Typisches P*rogrammier-Szenario (Programming-in-the-small)*

- Editor-Compiler Zyklus
- Editor-Compiler-Linker Zyklus
- Editor-Compiler-Linker-Ausführen Zyklus

“Eigentlich müsste es einfacher gehen”

Syntax-gesteuerte Editoren

Der Editor “versteht” die Sprache

- dadurch wird die Entwicklung neuer Applikationen beschleunigt
- In der Regel wird das GUI mit einem anderen aber integrierten Editor erstellt
  - das Tool ist integriert und funktioniert gleich, unabhängig von der Aufgabe
- Editoren verstehen z.T. unterschiedliche Sprachen
  - MS Developer Studio
  - Macromedia Web Tools
- Pretty-Printer
  - nette Darstellung des erstellten Programms

Source Level Debugger

Der Programmierer testet auf Source Level

- Core Dumps werden kaum benötigt
- Assembler Listings sind einsehbar
- Linker Listings werden automatisch erstellt
- Programmdokumentation basiert auf eingegebenem Text

Wunsch der Programmierer

- beim Source Level Testen sollte man interaktiv Änderungen eingeben können.

Programmier Workbench

Strukturediteren

- Interface Checking
- Betriebsystem Umfeld
- Online Dokumentation
- Source Level Debugger

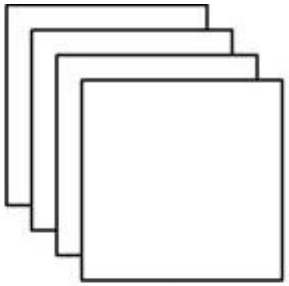
Ein Workbench stellt eine einfache Umgebung zur Verfügung

All dies ist schon sehr alt:

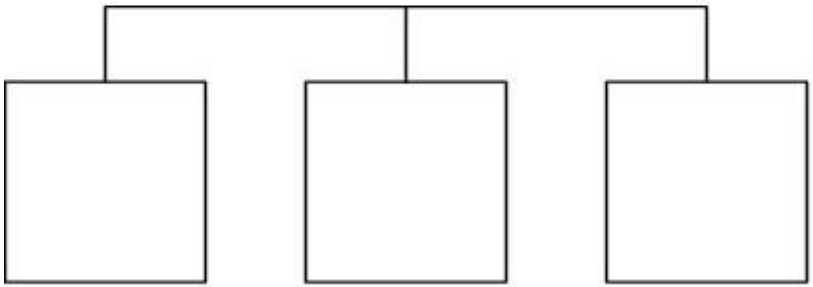
- FLOW Tool (1980)
- Technologie ist seit Jahren bekannt

Aber einige Programmierer lieben antike Werkzeuge

Revisionen und Variationen



(a)



(b)

Variation

- Version für unterschiedliche Betriebsysteme und / oder HW
- Variationen existieren gleichzeitig (eine Lsg wurde für mehrere Systeme Browser / ... entwickelt)

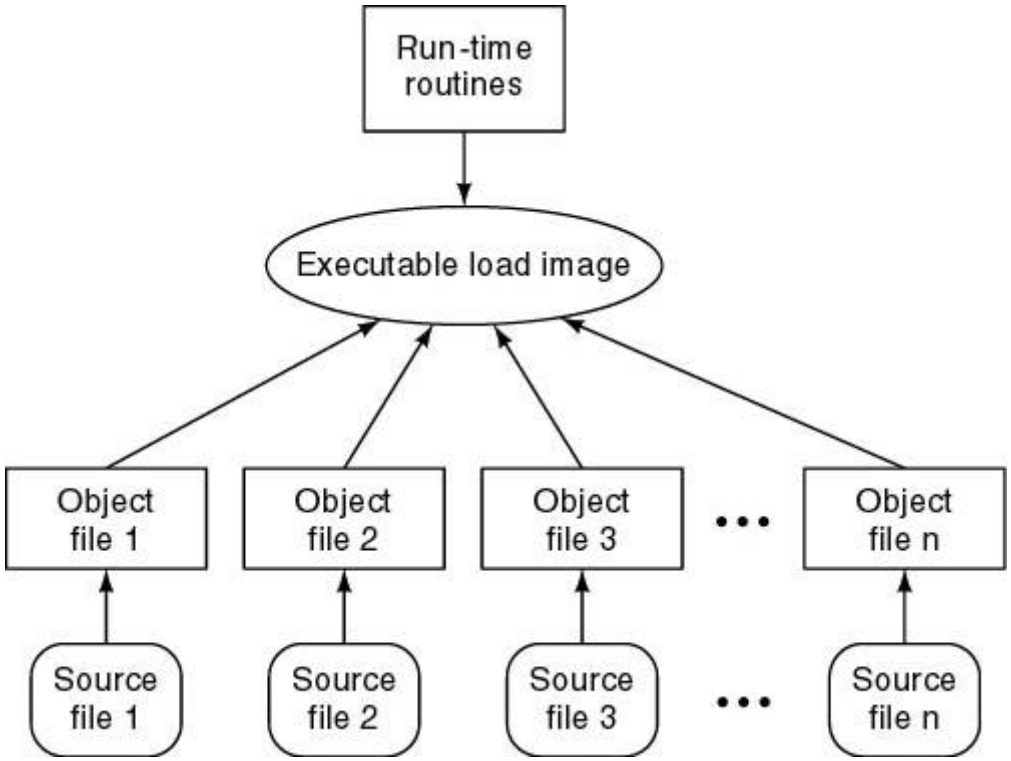
Konfigurations Kontrolle

Module existieren als

- Source Code;
- Objekt Code;
- ausführbare Exe's

Konfiguration

- jeder Modul wird versionsgerecht verwaltet



Build Tools

Beispiel

- UNIX make
- Apache ant

Vergleicht Zeitstempel und generiert neue Version

- praktisches Hilfsmittel zur Generierung auslieferbarer Software

Überprüft Abhängigkeiten

- und garantiert damit, dass Abhängigkeiten korrekt aufgelöst werden

Produktivitätsgewinn dank CASE Tools

Untersuchung von 45 Firmen in 10 Industrien [Myers, 1992]

- 50% kommerzielle Systeme
- 25% wissenschaftliche Software
- 25% Echtzeitsysteme (Flugüberwachung und ähnliches)

Resultate

- etwa 10% Produktivitätszuwachs pro Jahr
- $125,000 Kostenersparnis pro Jahr

Rechtfertigung von CASE

- beschleunigte Entwicklung
- weniger Fehler
- einfachere Wartung
- verbesserte Moral