

Chapter 5

Tools of the Trade!



5.1 Stepwise Refinement

- A basic principle underlying many software engineering techniques
 - “Postpone decisions as to details as late as possible to be able to concentrate on the important issues”
- Miller’s law (1956)
 - A human being can concentrate on 7 ± 2 items at a time



5.2 Cost– Benefit Analysis

- Compare costs and future benefits
 - Estimate costs
 - Estimate benefits
 - State all assumptions explicitly

Benefits		Costs	
Salary savings (7 years)	1,575,000	Hardware and software (7 years)	1,250,000
Improved cash flow (7 years)	875,000	Conversion cost (first year only)	350,000
		Explanations to customers (first year only)	125,000
Total benefits	\$2,450,000	Total costs	\$1,725,000

Figure 5.8


Cost–Benefit Analysis

Example: Computerizing KCEC



5.3 Divide-and-Conquer

- Solve a large, hard problem by breaking up into smaller subproblems that hopefully will be easier to solve
- Divide-and-conquer is used in the Unified Process to handle a large, complex system
 - Analysis workflow
 - Partition the software product into analysis *packages*
 - Design workflow
 - Break up the upcoming implementation workflow into manageable pieces, termed *subsystems*



5.4 **Separation of Concerns**

- The process of breaking a software product into components with minimal overlap of functionality
 - Minimizes regression faults
 - Promotes reuse
- Separation of concerns underlies much of software engineering



5.5 Software Metrics

- To detect problems early, it is essential to measure
- Examples:
 - LOC per month
 - Defects per 1000 lines of code

The Five Basic Metrics



Size

In lines of code, or better



Cost

In dollars



Duration

In months




Effort

In person months



Quality

Number of faults
detected



5.6 CASE (Computer- Aided Software Engineering)

- Scope of CASE
 - CASE can support the entire life-cycle
- The computer assists with drudge work
 - It manages all the details



Operating System Front- End in Editor

- Single command
 - go or run
 - Use of the mouse to choose
 - An icon, or
 - A menu selection
- This one command causes the editor to invoke the compiler, linker, loader, and execute the product



Source Level Debugger

- Example:
 - Product executes terminates abruptly and prints
Overflow at 4B06
 - or
Core dumped
 - or
Segmentation fault




Programming Workbench

- Structure editor with
 - Online interface checking capabilities
 - Operating system front-end
 - Online documentation
 - Source level debugger
- This constitutes a simple programming environment



5.9 Software Versions

- During maintenance, at all times there are at least two versions of the product:
 - The old version, and
 - The new version
- There are two types of versions: *revisions* and *variations*
- *Revisions are a change to the existing product*
- *Variations are changes made to have in run on new platforms*



5.9.1 Revisions

- Revision
 - A version to fix a fault in the artifact
 - We cannot throw away an incorrect version
 - The new version may be no better
 - Some sites may not install the new version
- Perfective and adaptive maintenance also result in revisions

5.10 Configuration Control

- Every code artifact exists in three forms
 - Source code
 - Compiled code
 - Executable load image
- Configuration
 - A version of each artifact from which a given version of a product is built

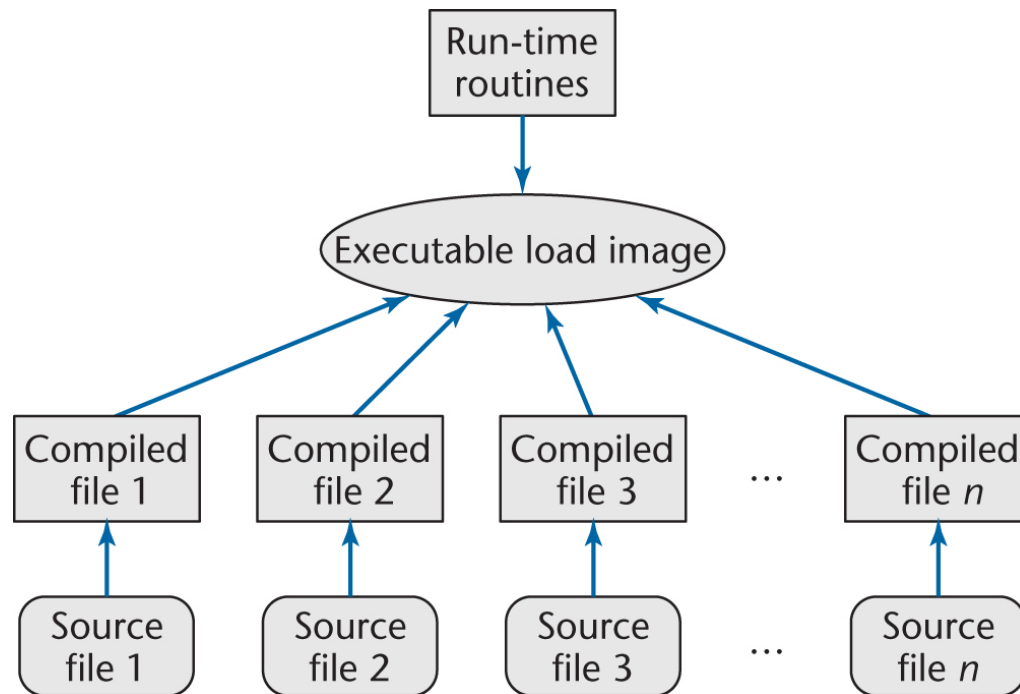


Figure 5.12



Version- Control Tool

- Essential for programming-in-the-many
 - **A first step toward configuration management**
- A version-control tool must handle
 - Updates
 - Parallel versions
 - branches



Version- Control Tools

- UNIX version-control tools
 - *sccs*
 - *rcs*
 - *cv*s
- Popular commercial configuration-control tools
 - PVCS
 - SourceSafe
- Open-source configuration-control tools
 - *cv*s
 - Subversion
 - git



5.11 Build Tools

- Example
 - UNIX *make*
- A build tool compares the date and time stamp on
 - Source code, compiled code
 - It calls the appropriate compiler only if necessary
- The tool then compares the date and time stamp on
 - Compiled code, executable load image
 - It calls the linker only if necessary



- The end!

