CASE workbenches

- ⊗ Software tools to support specific process phases

Objectives

- ⊗ To describe different types of CASE workbench
- ⊗ To discuss the notion of open and closed CASE workbenches
- ⊗ To describe the structure and components of design, programming and testing workbenches
- ⊗ To introduce meta-CASE tools for CASE workbench creation

Topics covered

- Programming workbenches
- Analysis and design workbenches
- Testing workbenches
- Meta-CASE workbenches

CASE workbenches

- A set of tools which supports a particular phase in the software process
- Tools work together to provide comprehensive support
- Common services are provided which are used by all tools and some data integration is supported

Types of workbench

- ⊗ Programming, design and testing workbenches covered here
- ⊗ Other types of workbench are
  - Cross-development workbenches for host-target development
  - Configuration management workbenches (discussed in Chapter 32)
  - Documentation workbenches for producing professional system documentation
  - Project management workbenches. Some management tools are discussed in Chapters 3 and 29

Open workbenches

- ⊗ Control integration mechanisms are provided and the data integration protocols are public. New tools can therefore be added by users
- ⊗ Advantages
  - The workbench can be tailored to specific organizational needs
  - The file outputs may be managed by a configuration management system
  - Incremental workbench introduction and evolution is possible
  - Organizations can source tools from different vendors. Diversity of supply is possible

Closed workbenches

- Many commercial workbenches are closed systems. The control and data integration protocols are proprietary. These are more common than open workbenches
- Allows for tighter tool integration including presentation integration
- However, it is impossible to integrate third-party tools and the user is tied to a single supplier

Programming workbenches

- A set of tools to support program development
- First CASE workbenches. Include compilers, linkers, loaders, etc.
- Programming workbenches are often integrated around an abstract program representation (the abstract syntax tree) which allows for tight integration of tools
- Integration around shared source-code files is also possible

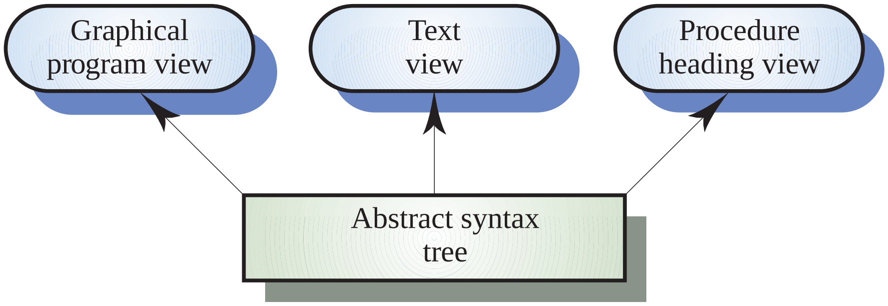
A programming workbench

- Replace with portrait slide

Language-directed workbenches

- Integrated around an abstract program representation
- The system editor has language knowledge and can edit the abstract representation rather than the source code text
- A range of program analysis tools may be supported
- Allow multiple views of the program to be generated

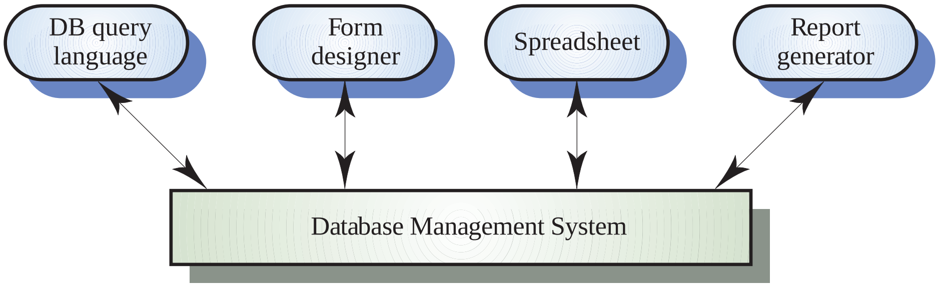
Multiple program views



4GL workbenches

- Provide facilities for developing 4GL programs
- Integrated around a database management system
- Components usually include
  - Database query language
  - Form design system
  - Spreadsheet
  - Report generator
- Very effective in developing business systems

A 4GL workbench



DB query language

Form designer

Spreadsheet

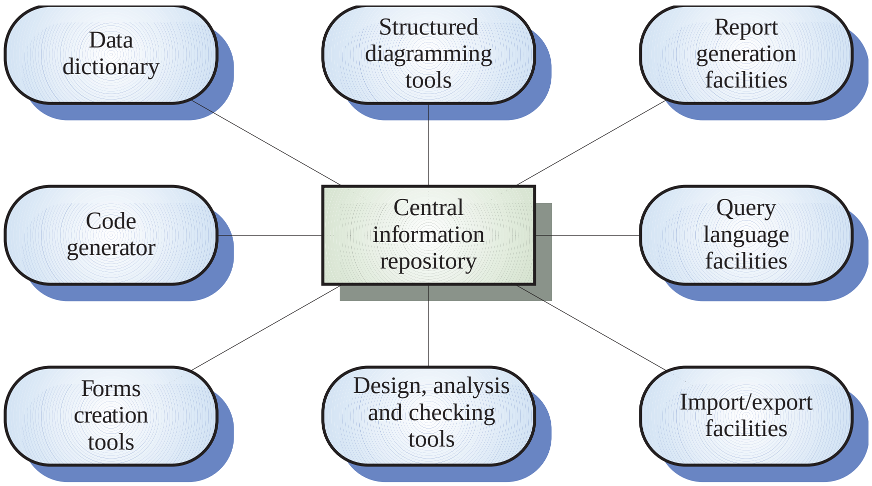
Report generator

Database Management System

Design and analysis workbenches

- Support the generation of system models during design and analysis activities
- Usually intended to support a specific structured method
- Provide graphical editors plus a shared repository
- May include code generators to create source code from design information

An analysis and design workbench


Data dictionary
Structured diagramming tools
Report generation facilities
Code generator
Central information repository
Query language facilities
Forms creation tools
Design, analysis and checking tools
Import/export facilities

Workbench advantages

- ⊗ Generally available on relatively cheap personal computers
- ⊗ Results in standardized documentation for software systems
- ⊗ Estimated that productivity improvements of 40% are possible with fewer defects in the completed systems

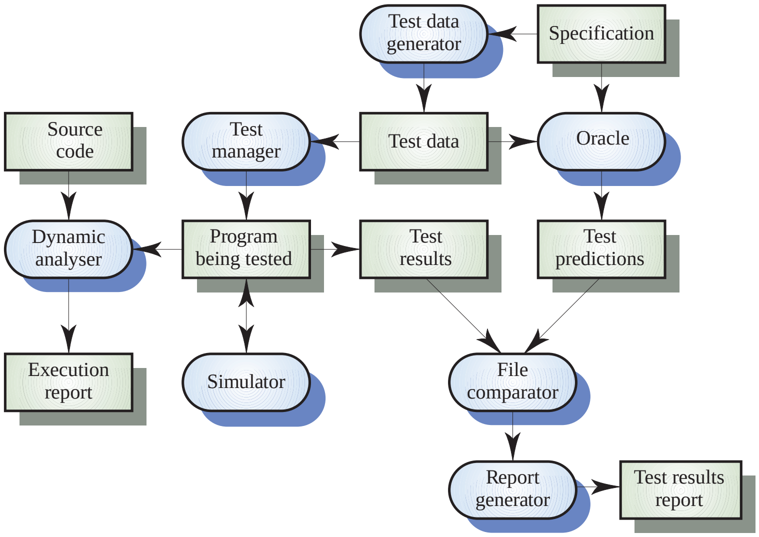
Workbench drawbacks

- These systems are usually closed environments with tight integration between the tools
- Import/export facilities are limited. ASCII and Postscript diagrams
- Difficult or impossible to adapt method to specific organizational needs
- Configuration management may either be excluded or specific to that workbench. Difficult to integrate with other systems in the organization

Testing workbenches

- Testing is an expensive process phase. Testing workbenches provide a range of tools to reduce the time required and total testing costs
- Most testing workbenches are open systems because testing needs are organization-specific
- Difficult to integrate with closed design and analysis workbenches

A testing workbench



Test data generator

Specification

Source code

Test manager

Test data

Oracle

Dynamic analyser

Program being tested

Test results

Test predictions

Execution report

Simulator

File comparator

Report generator

Test results report

Testing workbench adaptations

- ⊗ Scripts may be developed for user interface simulators and patterns for test data generators
- ⊗ Test outputs may have to be prepared manually for comparison
- ⊗ Special-purpose file comparators may be developed

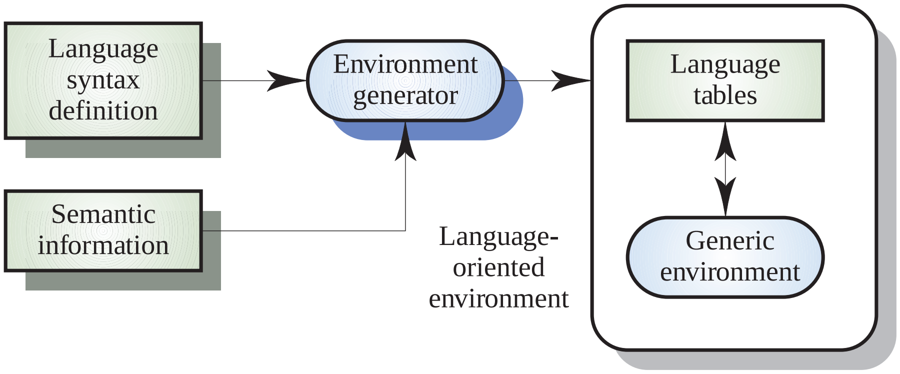
Meta-CASE

- Design and analysis workbenches are conceptually similar. Often the differences are only in the diagram types supported and the method rules and guidelines
- Programming workbenches are integrated around a syntax representation which may be separately defined
- Meta-CASE workbenches are tools which assist the process of creating workbenches. They reduce the costs of CASE workbench creation

Programming workbench generators

- First tools of this type were generated in the early 1980s (Mentor, Synthesizer Generator, Gandalf)
- The syntax and semantics of the programming language is defined and used to tailor generic language processing tools

Environment generation



Language syntax definition

Semantic information

Environment generator

Language-oriented environment

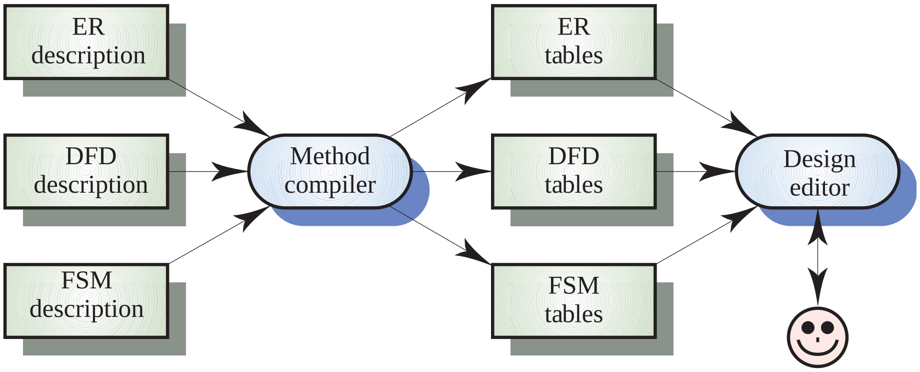
Language tables

Generic environment

Design workbench generation

- ⊗ Design and analysis workbenches can be created by using a method-definition language to define the method rules and guidelines
- ⊗ Components of a meta-CASE workbench include
  - General-purpose repository
  - Tools to create structure editors or textual notations and programming languages
  - A generic diagram editing system
  - Code generators for various languages
  - Forms and report generators

A multi-notation design editor



ER description

DFD description

FSM description

Method compiler

ER tables

DFD tables

FSM tables

Design editor

Key points

- CASE workbenches are integrated toolsets to support a phase of the software process
- Workbenches may be open or closed systems
- Programming workbenches, analysis and design workbenches and testing workbenches are widely used
- Analysis and design workbenches may include graphical editors, report generators and a data dictionary

Key points

- Testing workbenches may include test managers, dynamic analyzers, test data generators, file comparators and different types of emulator
- Meta-CASE workbenches are CASE systems which are used to generate other CASE systems. They may be based on descriptions of the notations and rules of design methods