Computer-aided software engineering

- Software tool support for software development

CASE technology

- Production-process support technology
  - Tools to support development activities such as specification, design, implementation, etc.
- Process management technology
  - Tools to support process modeling and management
- Meta-CASE technology
  - Generators used to produce CASE toolsets

Impact of CASE technology

- CASE technology has resulted in significant improvements in quality and productivity
- However, the scale of these improvements is less than was initially predicted by early technology developers
  - Many software development problems such as management problems are not amenable to automation
  - CASE systems are not integrated
  - Adopters of CASE technology underestimated the training and process adaptation costs

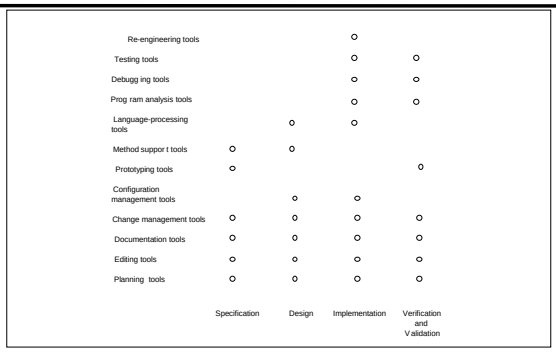
CASE classification

- CASE systems can be classified according to their
  - Functionality - what functions do they provide
  - Process support - what software process activities do they support
  - The breadth of support which they provide
- Classification allows tools to be assessed and compared

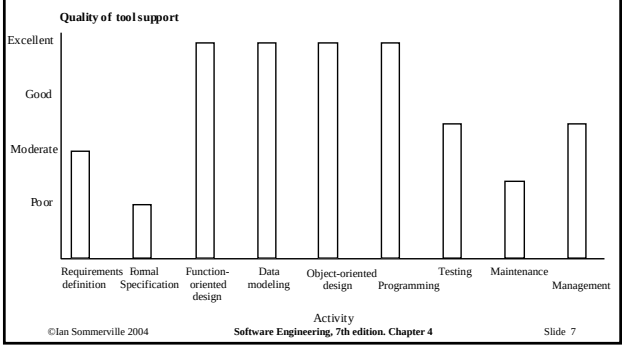
Functional tool classification

Tool type	**Examples**
Planning tools	PERT tools, estimation tools, spreadsheets
Editing tools	Text editors, diagram editors, word processors
Change management tools	Requirements traceability tools, change control systems
Configuration management tools	Version management systems, system building tools
Prototyping tools	Very high-level languages, user interface generators
Method-support tools	Design editors, data dictionaries, code generators
Language-processing tools	Compilers, interpreters
Program analysis tools	Cross reference generators, static analysers, dynamic analysers
Testing tools	Test data generators, file comparators
Debugging tools	Interactive debugging systems
Documentation tools	Page layout programs, image editors
Re-engineering tools	Cross-reference systems, program re-structuring systems

Activity-based tool classification



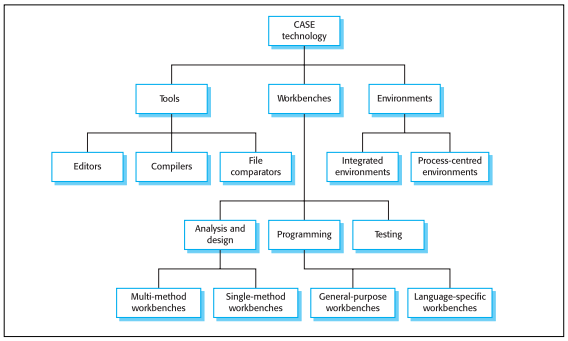
Quality of CASE support



CASE integration

- Tools
  - Support individual process tasks such as design consistency checking, text editing, etc.
- Workbenches
  - Support a process phase such as specification or design, Normally include a number of integrated tools.
- Environments
  - Support all or a substantial part of an entire software process. Normally include several integrated workbenches.

Tools, workbenches, environments



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
- Other types of workbench are
  - Cross-development workbenches for host-target development
  - Configuration management workbenches Documentation workbenches for producing professional system documentation
  - Project management workbenches.

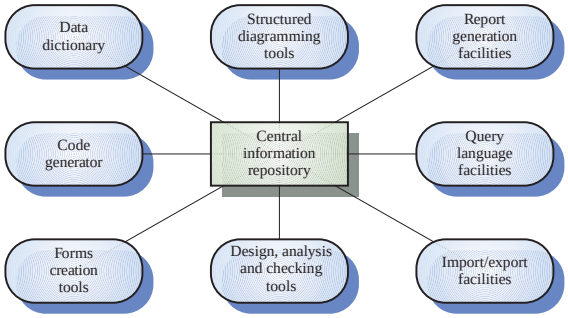
Programming workbenches

- 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

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

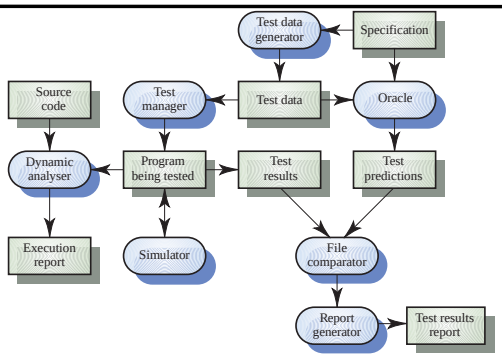
An analysis and design workbench



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



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

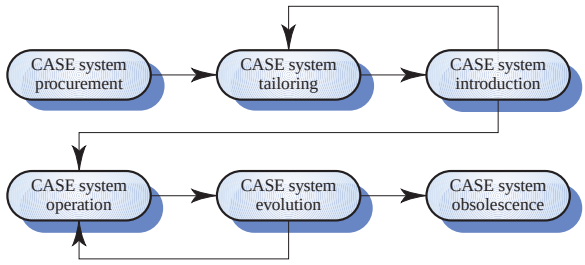
Meta-CASE

- 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

The CASE life cycle

- Procurement
- Tailoring
- Introduction
- Operation
- Evolution
- Obsolescence

A CASE life cycle model



CASE procurement

- Existing company standards and methods
  - The environment must support existing practice
- Existing and future hardware
  - The environment must be compatible with existing hardware. It should run on industry-standard machines
- The class of application to be developed
  - The environment should support the principal type of application developed by an organization
- Security
  - The environment should provide appropriate access control facilities

CASE system tailoring

- Installation
  - Set system dependent hardware and software parameters
- Process model definition
  - Define the activities that the environment is to support
- Tool integration
  - Describe what tools are to be part of the environment and how they are to be integrated
- Documentation
  - Provide appropriate, in-house documentation for using the environment

CASE introduction and operation

- May require changes to working practice
  - User resistance because of conservatism or a feeling that environments are for managers rather than engineers
  - Lack of training. Organisations often don't invest enough in training
  - Management resistance. Managers may not see how the environment will reduce project costs
- Migrate projects slowly to the CASE system
  - New projects should start with the environment after initial pilot projects have demonstrated its advantages
  - It is usually impractical to convert existing projects to the CASE system

CASE system evolution

- As the system is used, new requirements arise
  - Process requirements. Changes in the process model will be identified
  - Tool requirements. New tools will become available and will have to be incorporated
  - Data requirements. The data organisation will evolve
- An evolution budget must be available or the environment will become progressively less useful
- Forward compatibility must be maintained

CASE system obsolescence

- At some stage, an environment will outlive its usefulness and will have to be replaced
- Replacing an environment must be planned and should take place over an extended time period
- Currently supported projects must be moved to a new environment before their supporting environment is scrapped