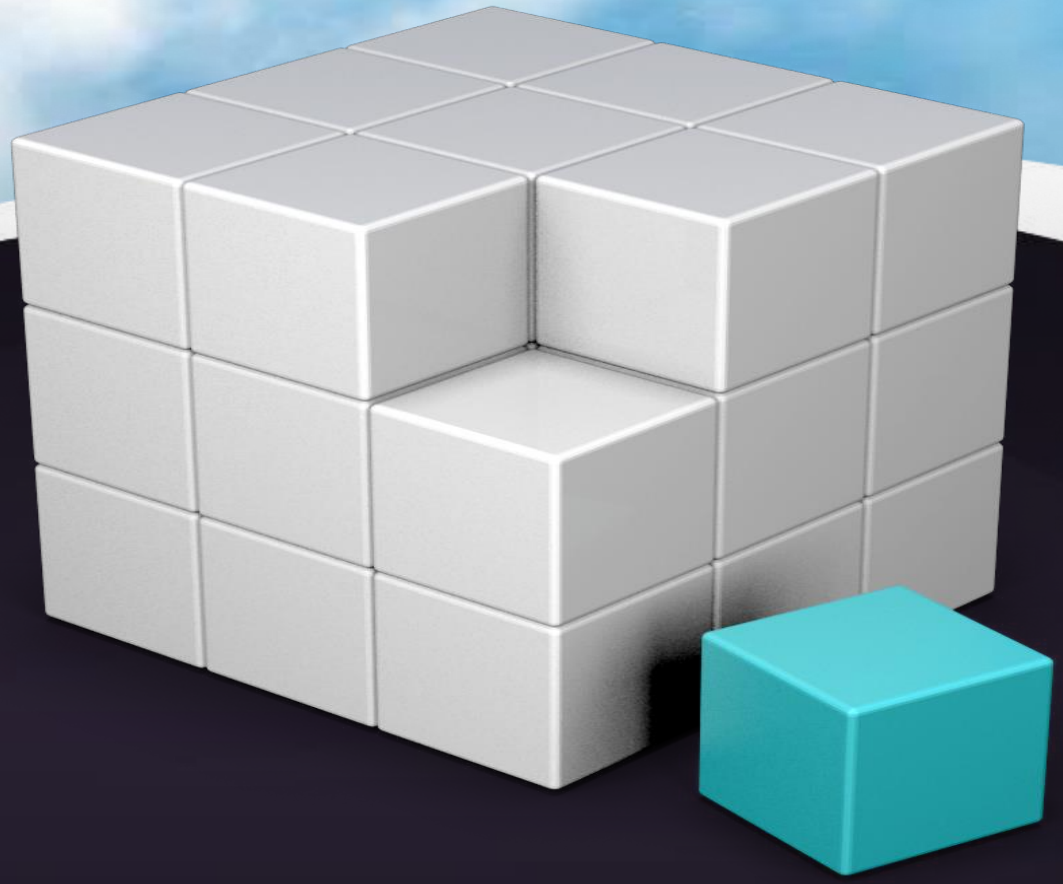
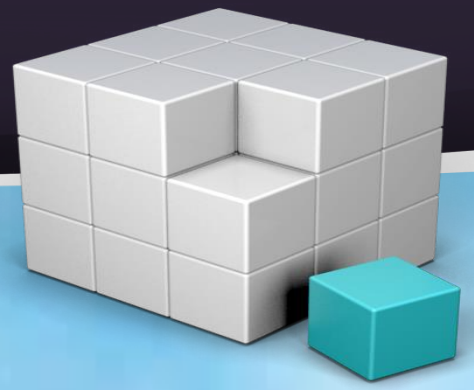
Component-Based Software Engineering (CBSE) & Computer-Aided Software Engineering (CASE)

Chapter 12

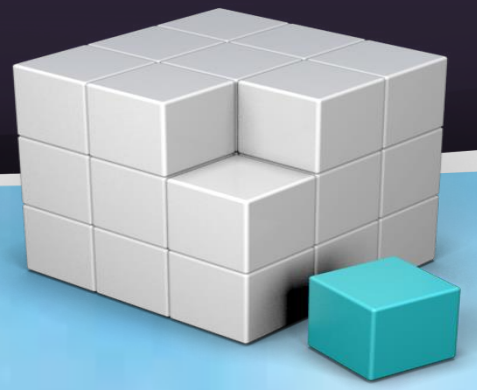


Lesson Objectives



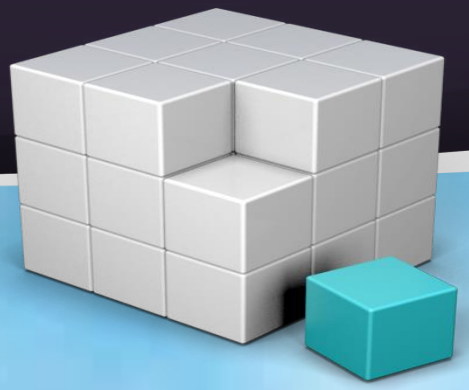
- Explain the 4 main activities in CBSE process
- Evaluate the use of CBSE approach – Benefits and Obstacles

Introduction



- Component-based software engineering (CBSE) is a process that emphasizes the ***design*** and ***construction*** of computer-based systems using ***reusable software "components".***
- ***E.g.:***
  - ***Reuse the programs / screens from previous projects/ Assignments***

Introduction



Establish SRS,
Architectural design

Examine any components **can be reused**?

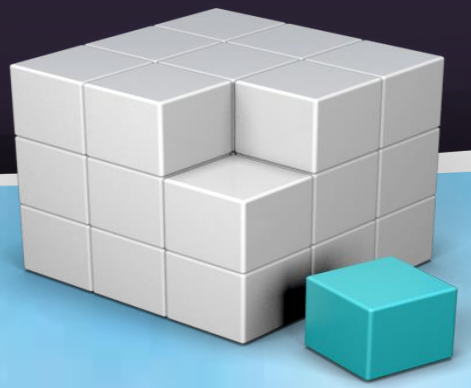
Yes

No

CBSE

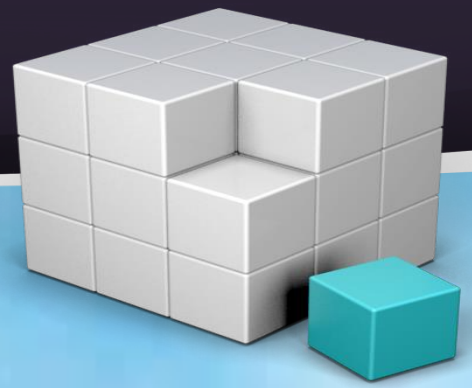
- *Component Qualification*
- *Component Adaptation*
- *Component Composition*
- *Component Update*

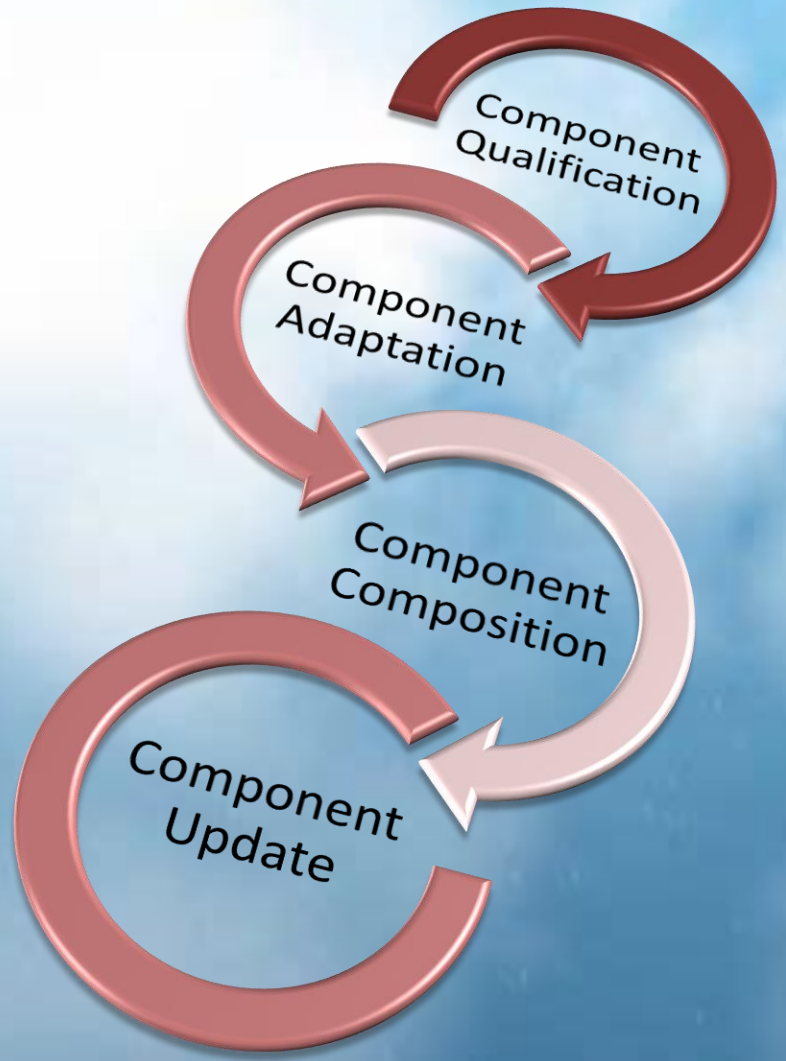
Use conventional or O-O SE to develop new components



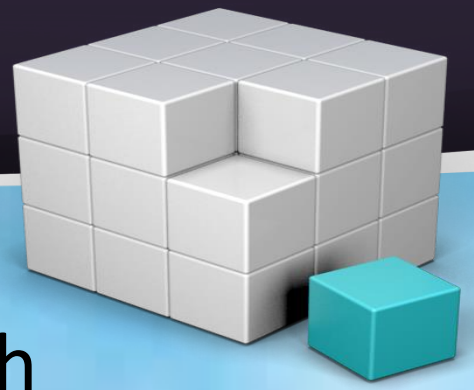
4 Activities in CBSE

CBSE Activities



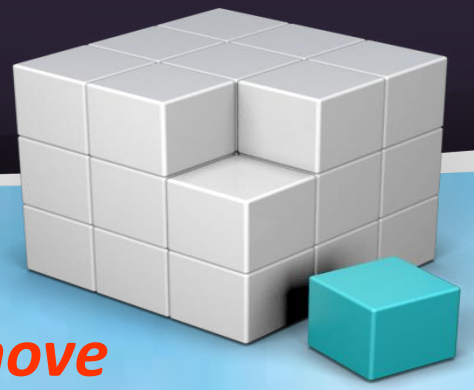


CBSE Activities – Component Qualification



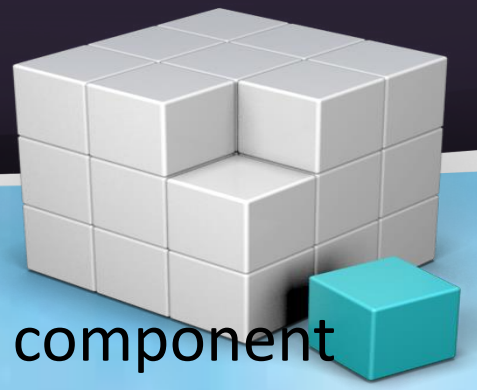
- Use a process of discovery and analysis ***to qualify*** each component's fit in ***architecture and requirements***.
- Ensures that a candidate component will perform the function required
- Before a component can be used, factors need to consider includes: the run-time requirements, services requirements, exception handling, security features etc

CBSE Activities – Component Adaptation



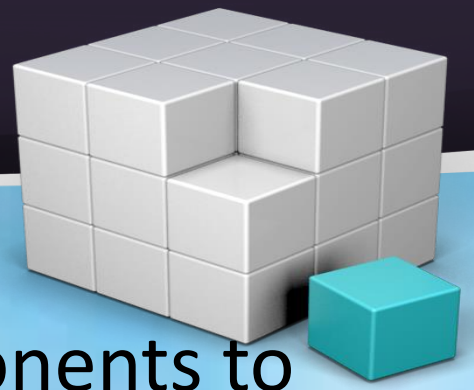
- Components adapted to ***meet the needs of architecture*** or to ***remove*** architectural mismatches, and be ***replaced*** by more suitable components
- In reality, qualified component for use may exhibit conflict in one or more of the areas.

CBSE Activities – Component Adaptation



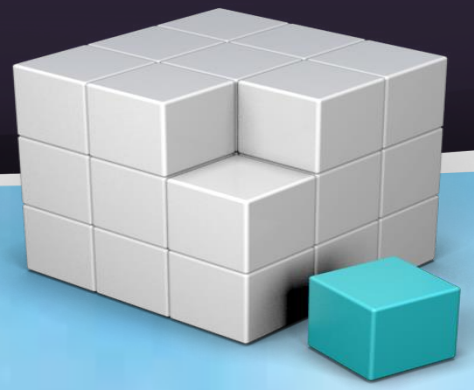
- To mitigate against these conflicts, an adaptation technique called component wrapping is often used.
- ***white-box wrapping***
  - examines the internal processing details of the component and makes code-level modifications to remove any conflict.
  - Not for the case when COTS components are used.
- ***gray-box wrapping***
  - applied when the component library provides a component extension language or API that enables conflicts to be removed or masked.
- ***black-box wrapping***
  - requires the introduction of pre- and post processing at the component interface to remove or mask conflicts.

CBSE Activities – Component Composition

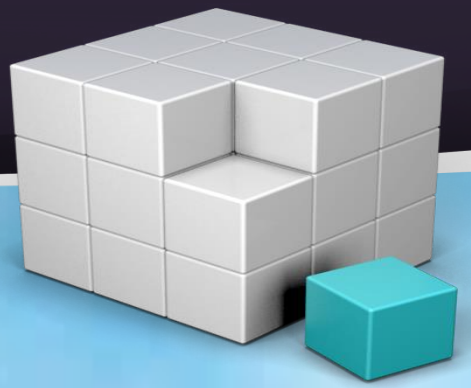


- ***Assembles*** qualified, adapted, and engineered components to populate the architecture established for an application.
- An infrastructure must be established to bind the components into an operational system.

CBSE Activities – Component Update

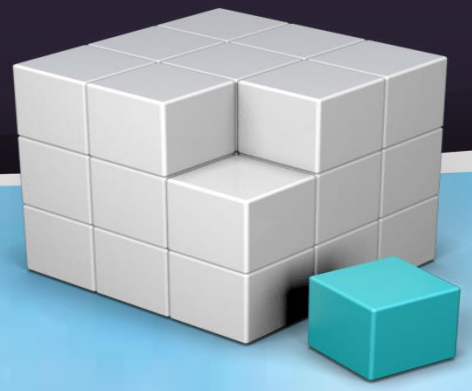


- When requirements for the system change
- ***Update*** the components if requirements change / new release available



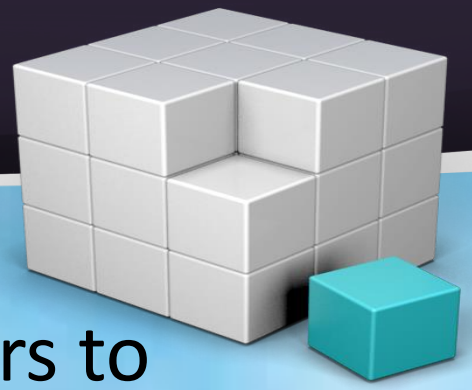
Benefits & Obstacles of CBSE

Evaluation – Benefits of CBSE

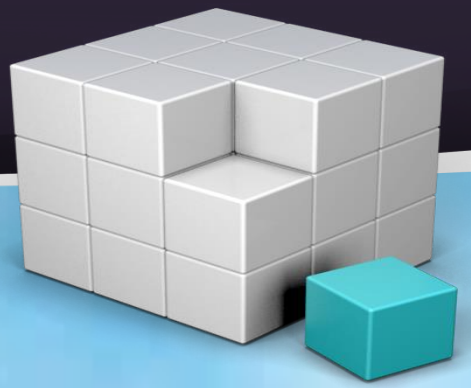


- ***Improve Quality***
  - provides a high-performance & more reliable end product as components have been used many times, each time getting more refined and reliable as they are reused and re-tested.
- ***Increase Productivity***
  - reduces the time required for code development, documentation, system modeling, and planning
- ***Save Cost***
  - reduces development cost from scratch & testing overhead

Evaluation – Obstacles of CBSE



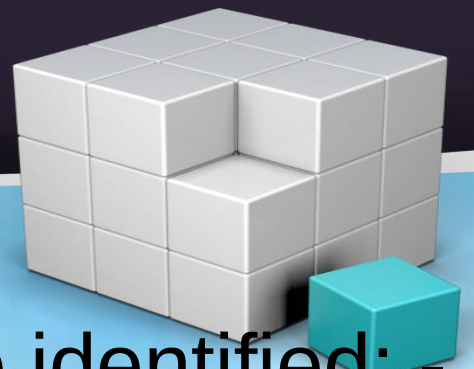
- ***Little training*** is available to help s/w engineers & mgrs to understand & apply resue
- Many s/w practitioners continue to believe that reuse is "***more trouble than it's worth***".
- ***Few*** s/w companies to ***provide incentive*** to reusable components program
- Many companies continue to ***encourage*** of ***s/w development methodologies which do not facilitate reuse.***



3 Level of CASE

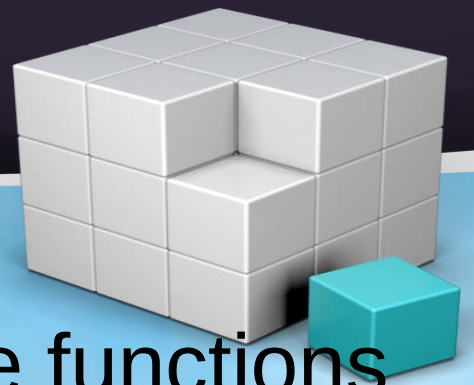
COMPUTER-AIDED SOFTWARE ENGINEERING (CASE) TECHNOLOGY

CASE Technology



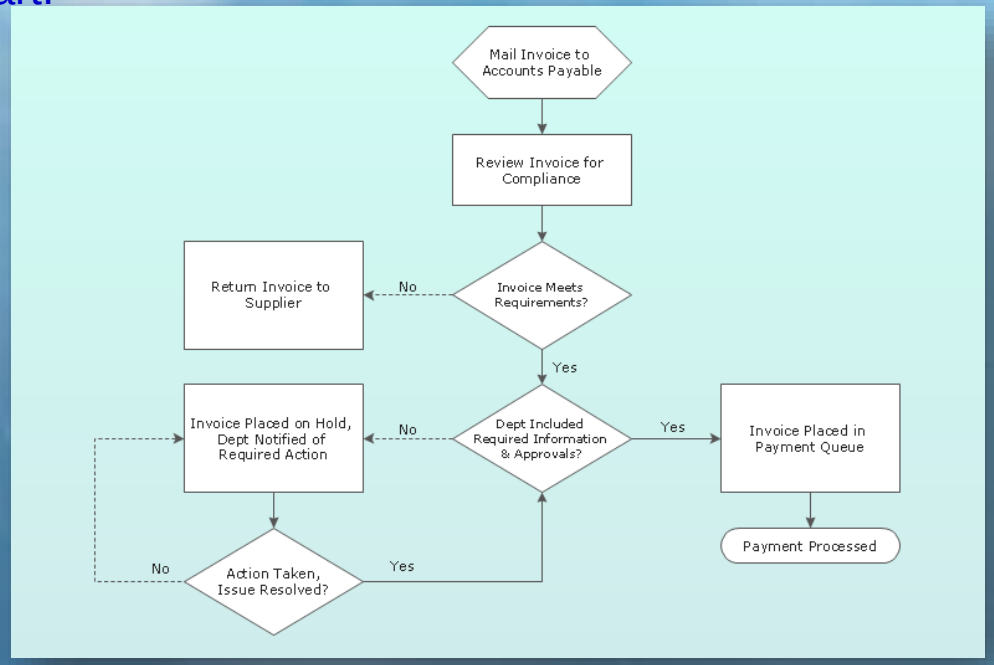
- Three different levels of CASE technology can be identified: -
  - ***Production-process support technology*** – support for process activities such as specification, design, testing and etc.
  - ***Process management technology*** – support modeling and process management.
  - ***Meta-CASE technology*** – generators which are used to create production-process and process management support tools.

Functional Classification of CASE

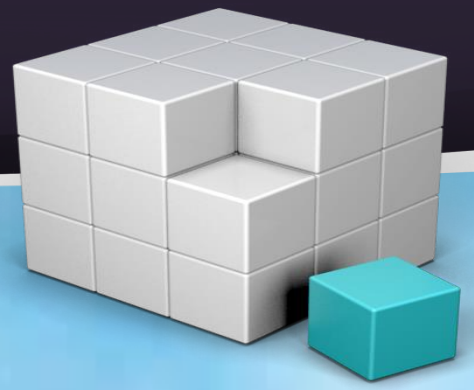


- **Functionality**-tools are specified according to the functions they support:

  1. Business process engineering tools – trace and track data flows between different original departments. Output of this CASE tool can be a flowchart.



Functional Classification of CASE

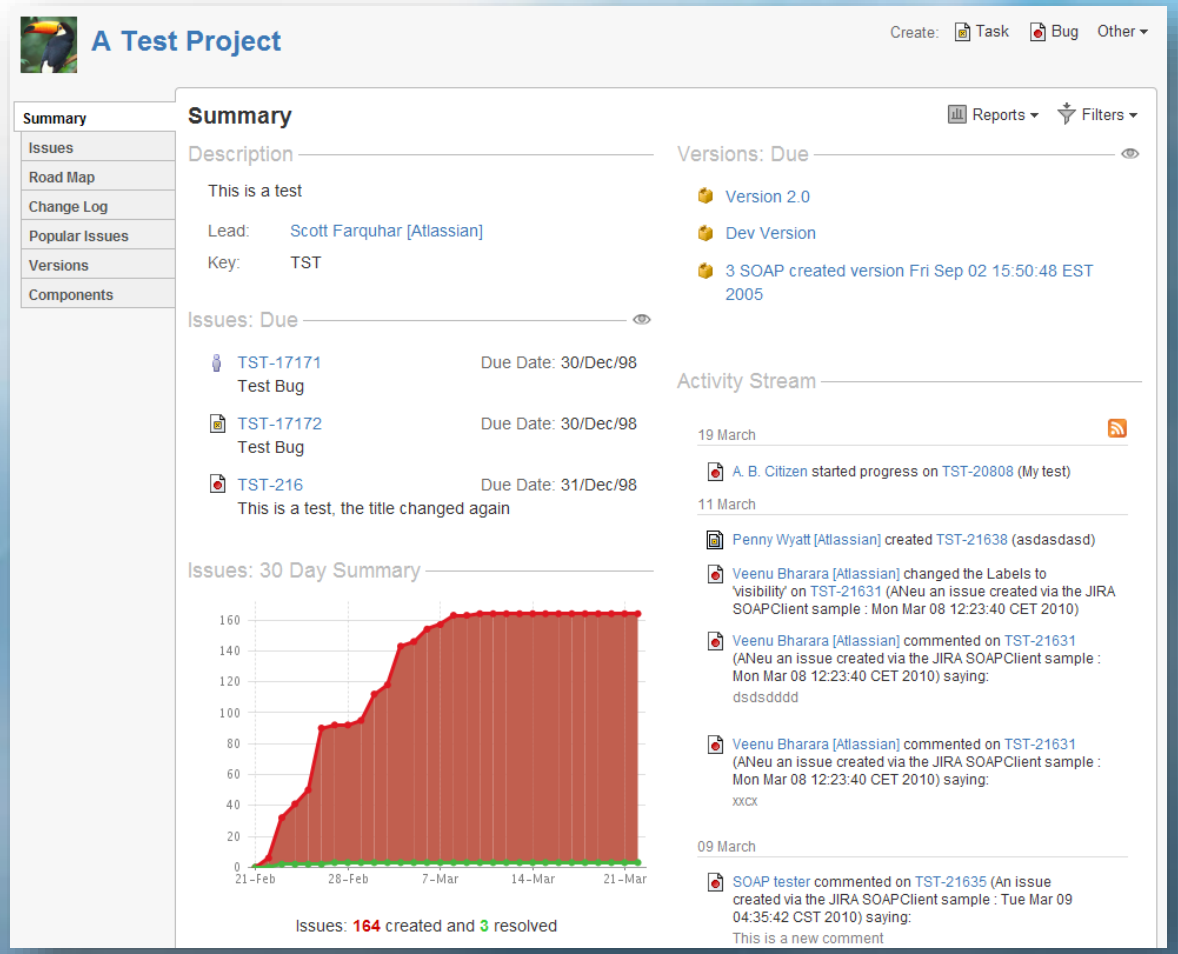


2. Process modeling & management tools – e.g. **Visio**

3. Data modeling tools – DBMS, etc

Functional Classification of CASE

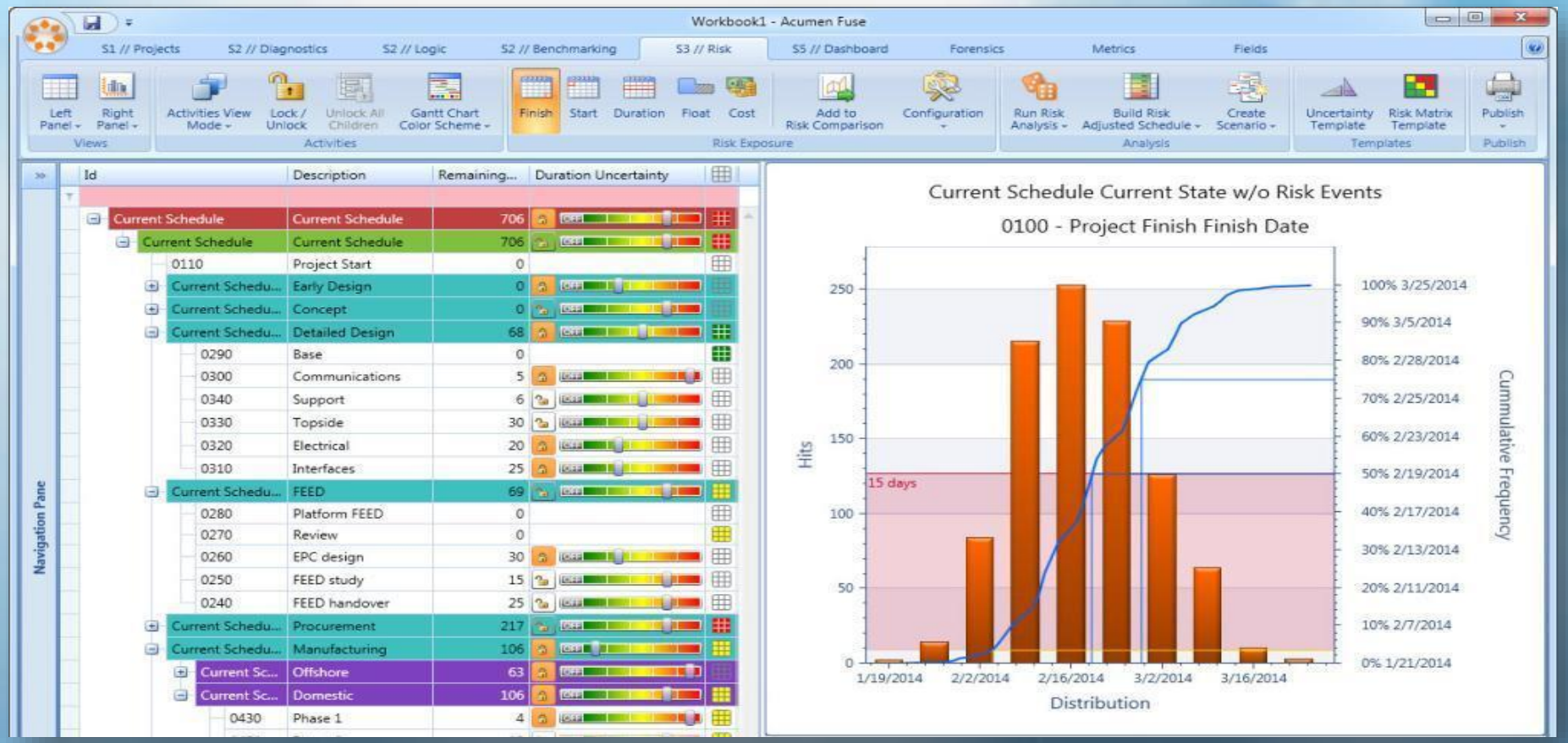
4. Project planning/Project management tools – cost /cost and time estimation, e.g. Excel Spreadsheet



Atlassian JIRA project management tool

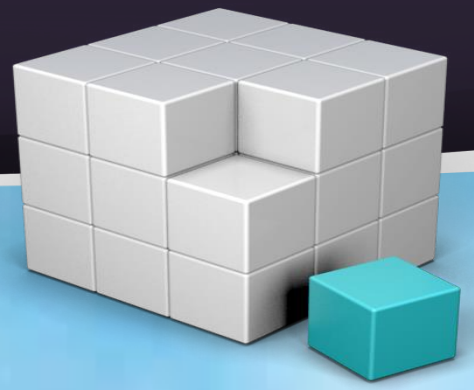
Functional Classification of CASE

5. Risk Analysis Tools – manage project risk

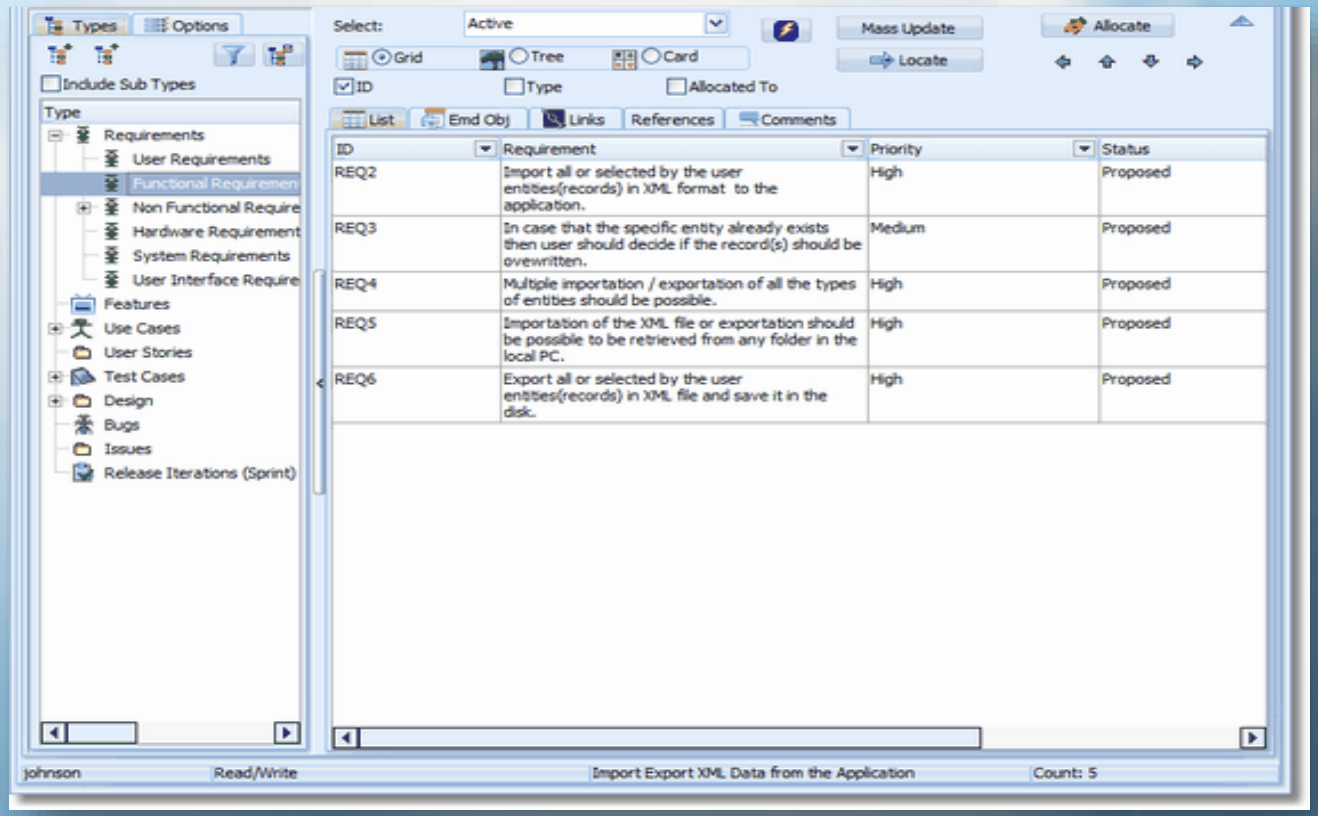


Acumen Risk is a Monte Carlo risk analysis tool combining true cost and schedule risk analysis against a native project plan together with identified risk events from a project risk register.

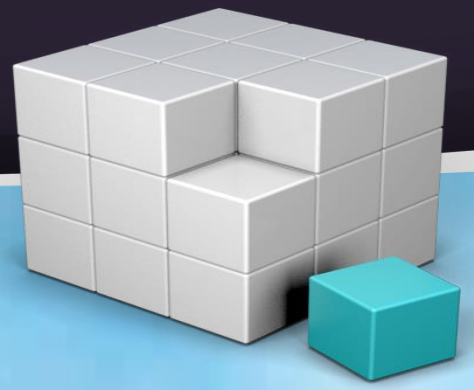
Functional Classification of CASE



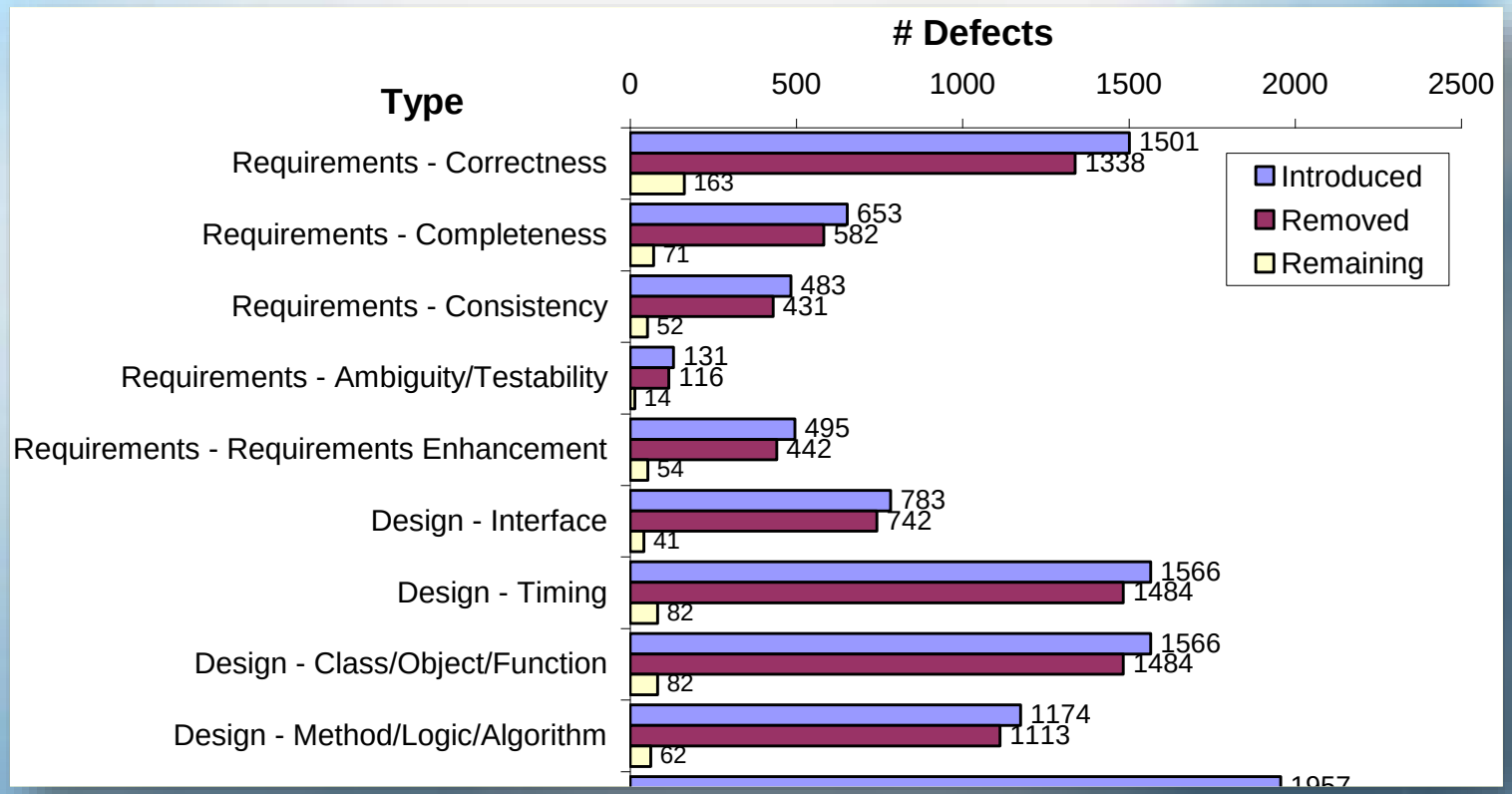
6. Requirement Tracing Tools – to ensure functions delivered completely as stated



Functional Classification of CASE

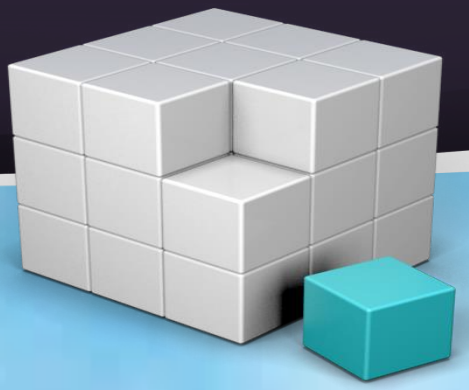


7. Metrics & Management Tools – for software quality measurement



COQUALMO is a tool used to predict the number of residual defects in a software product.

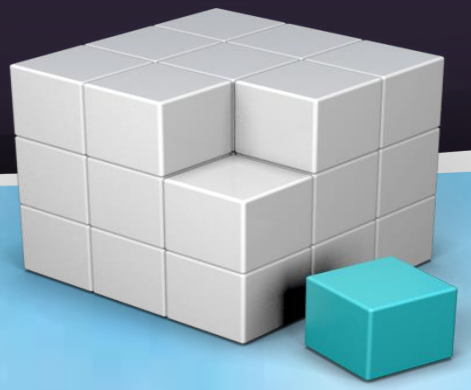
Functional Classification of CASE



8. Prototyping Tools - user interface generators, e.g. VB

UI generator

Functional Classification of CASE



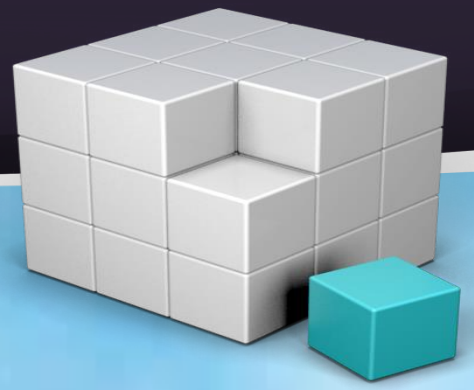
9. Editing Tools - text & diagram editors

10. Method support tools – design & code editors, data dictionaries, e.g. IBM Rational Software Architect (RSA)

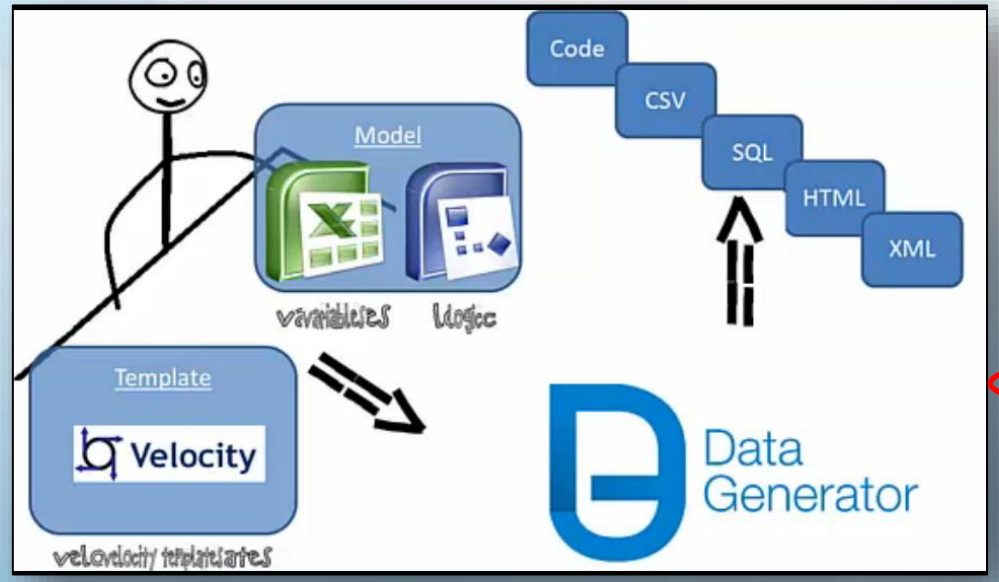
11. Language processing tools – compilers, interpreters

12. Debugging tools – interactive debugging systems

Functional Classification of CASE

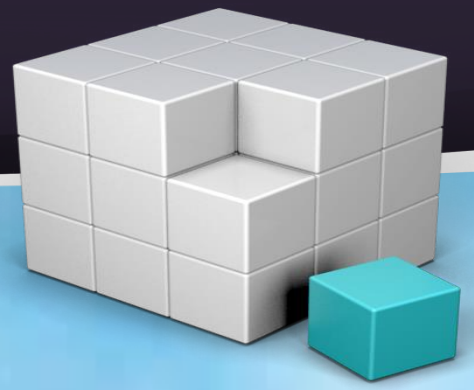


13. Testing Tools - test data generators, test coverage analysis and reporting

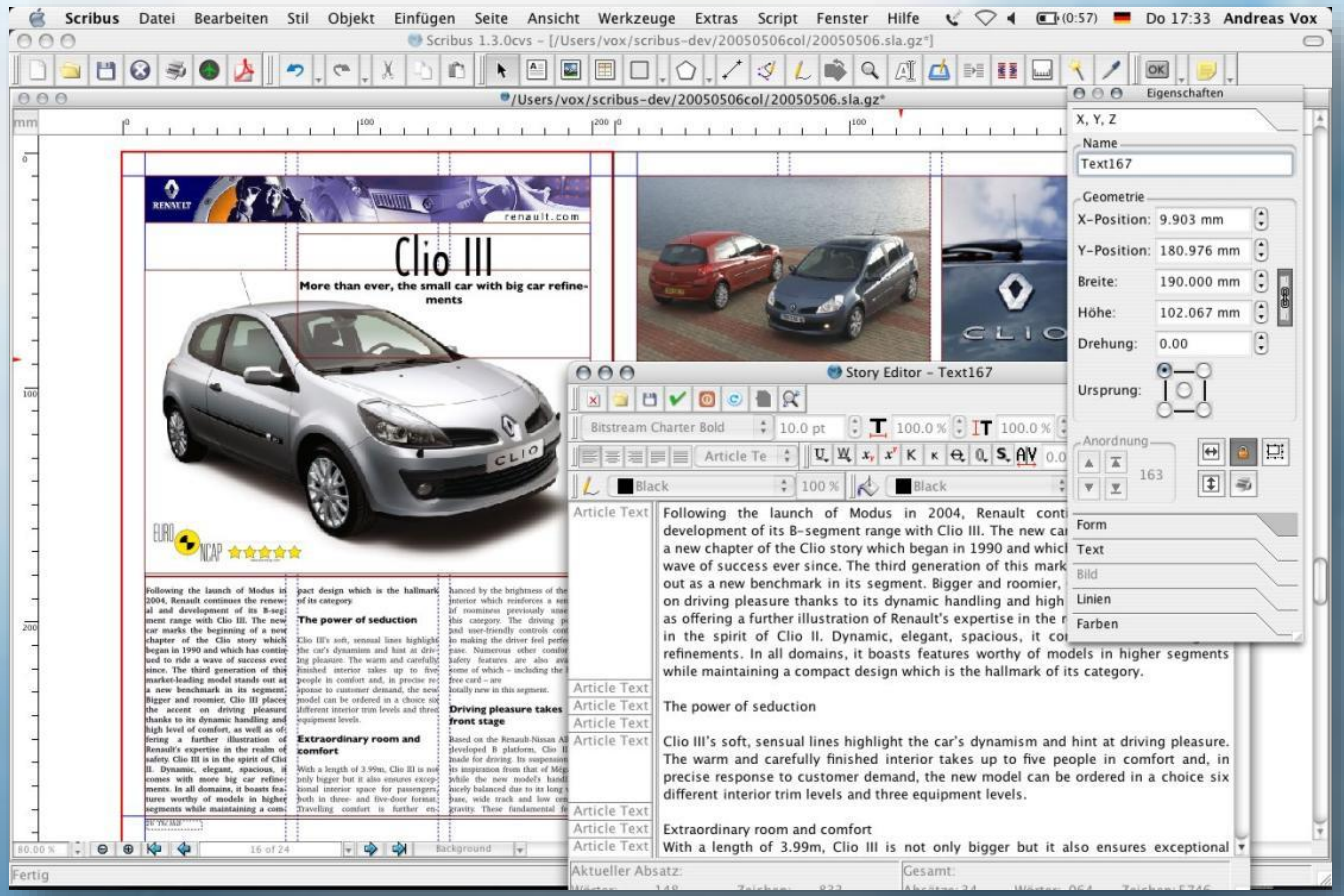



```
                    Semantic Designs Test Coverage Report


Probe Reference File:
      \\Nyx\C\DMS\Analyzers\GlobalPointerAnalysis\GlobalPointerAnalysis.prf

Test Coverage Vectors:
      \\Nyx\C\DMS\Analyzers\GlobalPointerAnalysis\%TestCoverage_2009_09_24_17_53_45_000.tcv

SUMMARY:
      Total Probes: 2169
      Total Files: 28
      52.7% covered (1145 out of 2169).
      47.2% uncovered (1024 out of 2169).

COVERAGE REPORT BY FILE:
      [1] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/CommandLine/CommandLine.h    88.8% uncovered (32 out of 36).
      [2] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/CommandLine/SequenceOfFiles.h    23.0% uncovered (3 out of 13).
      [3] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/File/Exceptions.h    100.0% uncovered (20 out of 20).
      [4] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/File/ReadUnicodeUTF16File.h    58.3% uncovered (14 out of 24).
      [5] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/File/WriteUnicodeUTF16File.h    55.1% uncovered (16 out of 29).
      [6] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/GlobalPointerAnalysis.cpp    42.4% uncovered (14 out of 33).
      [7] C:/DMS/Analyzers/GlobalPointerAnalysis/Source/InclusionConstraintGraph/CachedInformation.h    30.0% uncovered
```

Functional Classification of CASE

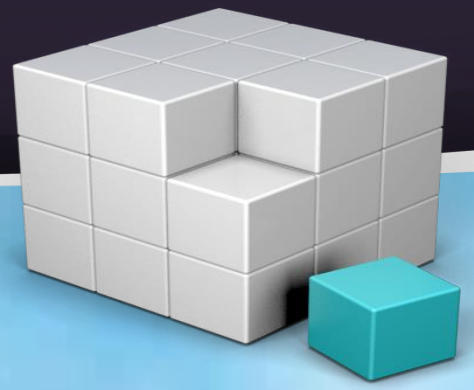


14. Documentation Tools - page layout programs, image editor

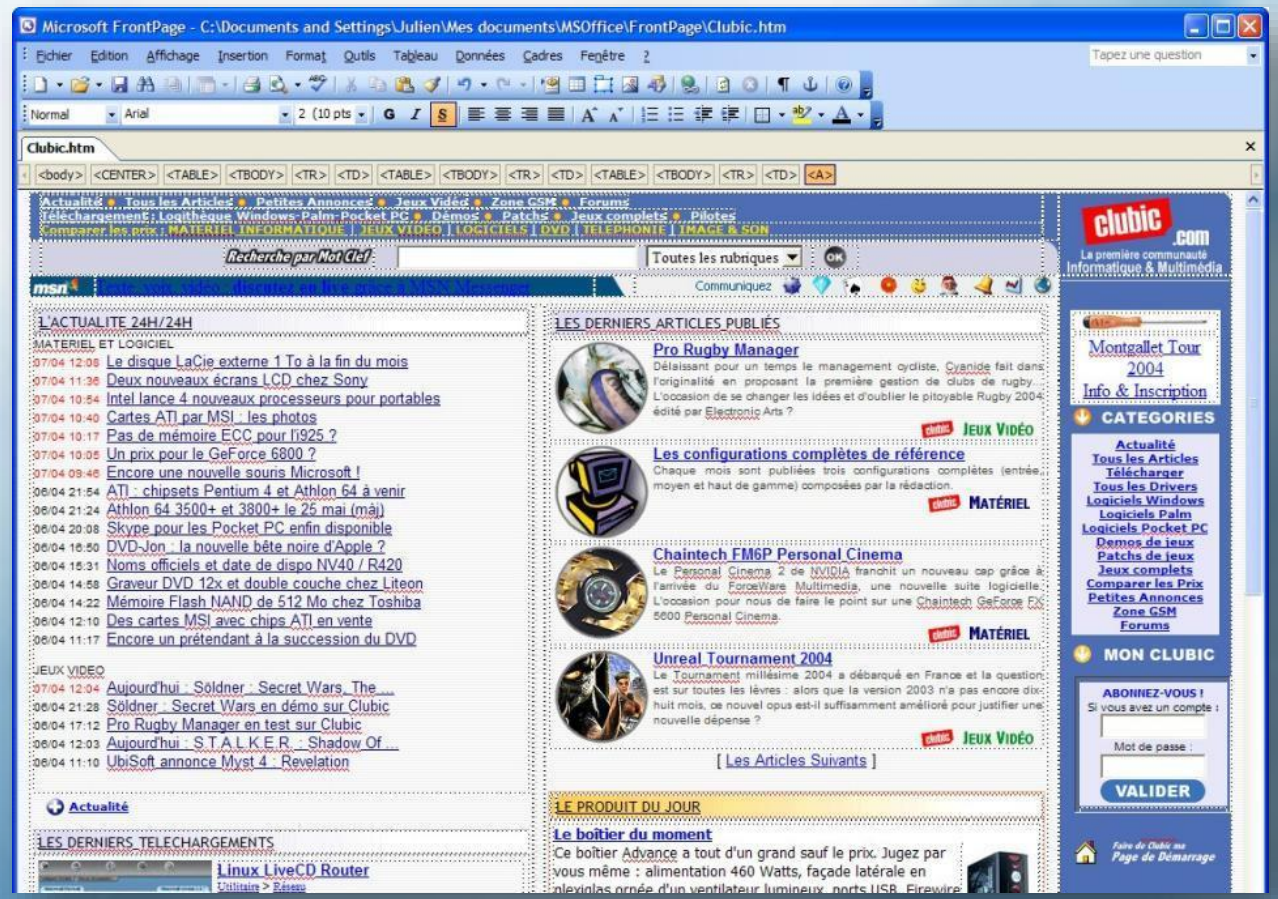


Scribus – page layout program

Functional Classification of CASE

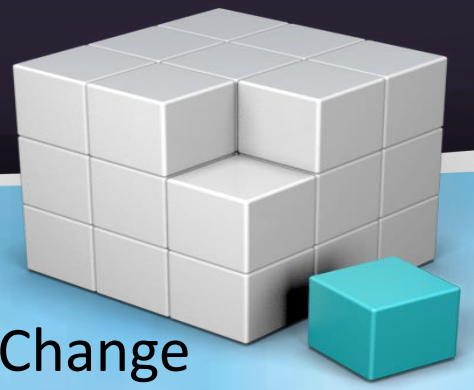


15. Web Development Tools



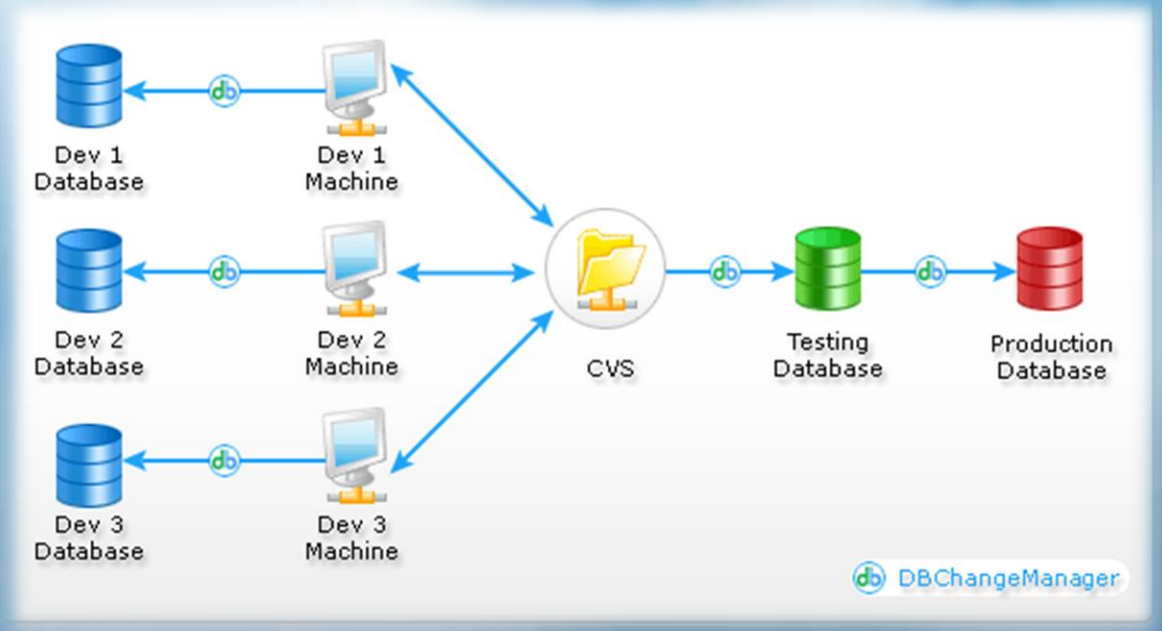
e.g. FrontPage

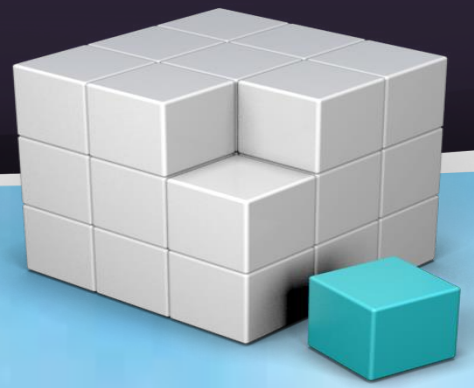
Functional Classification of CASE



16. Software Configuration Management Tools - e.g. SQL Server DB Change Management, Version & Change Management

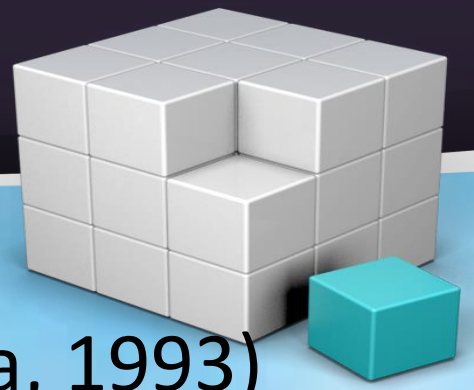
SQL Server Database Change Management:



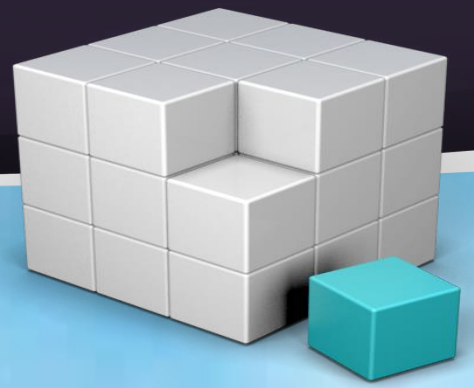


OTHER CLASSIFICATION OF CASE

Other classification of CASE

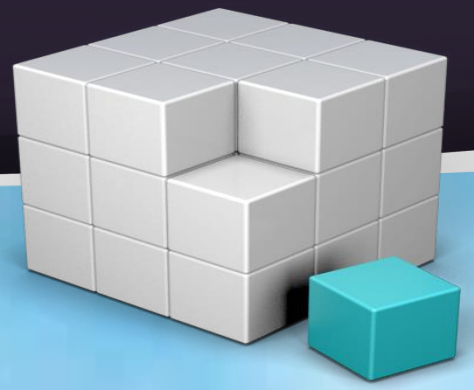


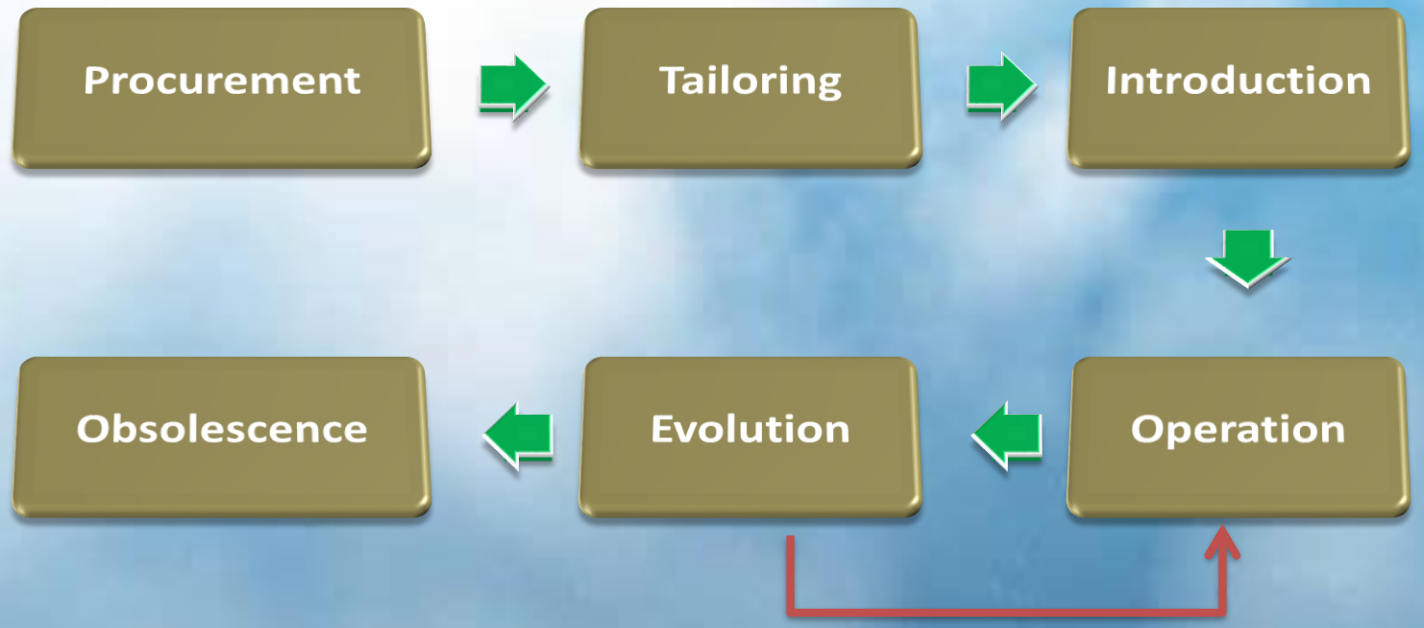
- Can also be classified by ***Breadth of Support*** (Fuggetta, 1993)
  - **Tools** – support individual tasks (stand alone tool) like compiler, word processor and etc
  - **Workbenches** – support process phases or activities like specification, analysis, design and etc. Normally, include a few tools.
  - **Environment** – support all or at least a substantial part of s/w process. Normally, include several workbenches that integrated together.



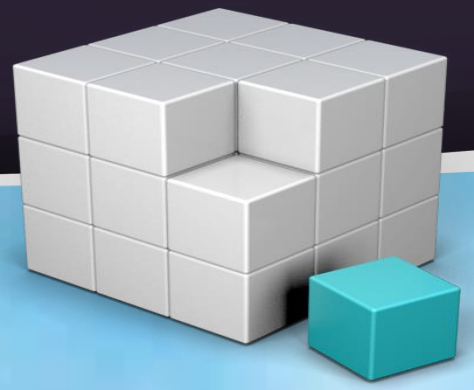
CASE LIFE CYCLE

CASE Life Cycle



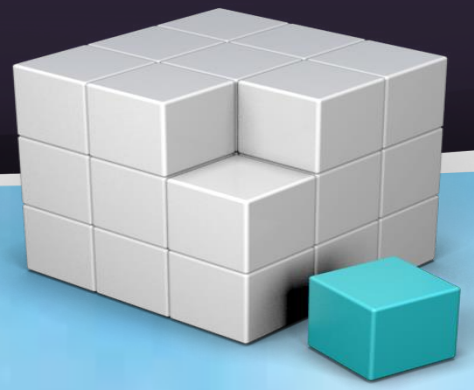


CASE Life Cycle

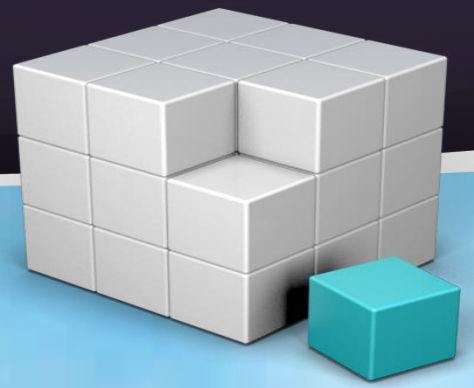


- The stages of the CASE life cycle are:-
  - ***Procurement*** – choose an appropriate CASE system
  - ***Tailoring*** – adapts a CASE system to a particular set of organization or project requirements
  - ***Introduction*** – introduces the CASE system into a working context

CASE Life Cycle

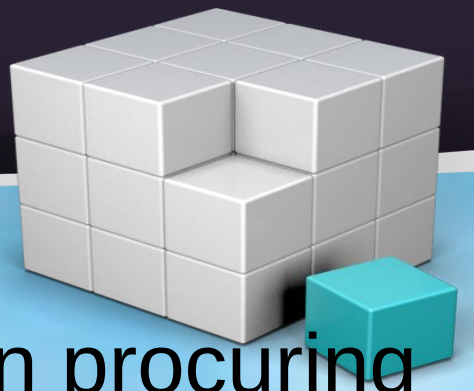


- The stages of the CASE life cycle are:-
  - ***Operation*** – the CASE system is in everyday use for software development
  - ***Evolution*** – modifying the h/w or s/w to adapt to new requirements
  - ***Obsolescence*** – the CASE is taken out of use. Need to ensure that the s/w developed using the system can still be supported by the org.



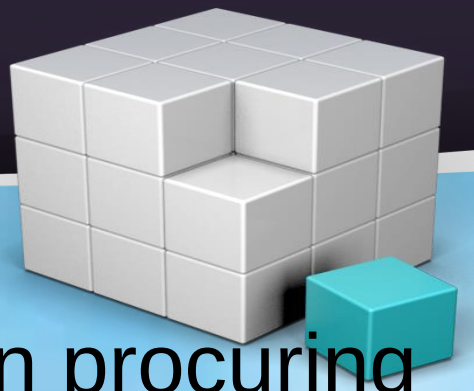
ISSUES IN CASE LIFE CYCLE

Issues in CASE Life Cycle – 1. Procument



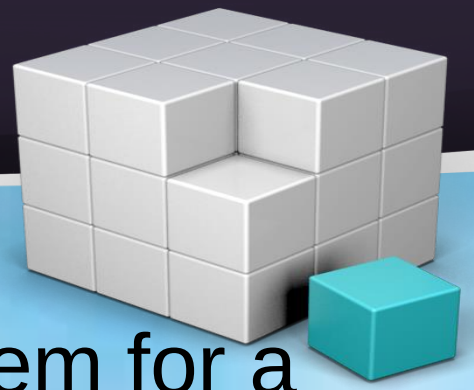
- The factor which must be taken into account when procuring a company-wide CASE system are: -
  - Existing company standards and methods – Visio supports SSADM convention?
  - Existing computers and future computer procurement
    - memory, CPU, OS

Issues in CASE Life Cycle – 1. Procument



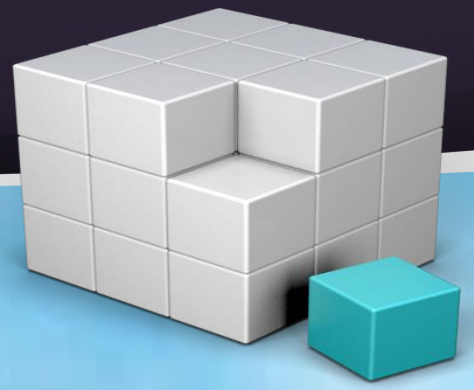
- The factor which must be taken into account when procuring a company-wide CASE system are (cont'): -
  - The class of applications to be developed – choose a CASE that provides facilities for developing the type of applications required by an org.
  - Security – provide any security features?
  - Cost – expensive? Within budget?

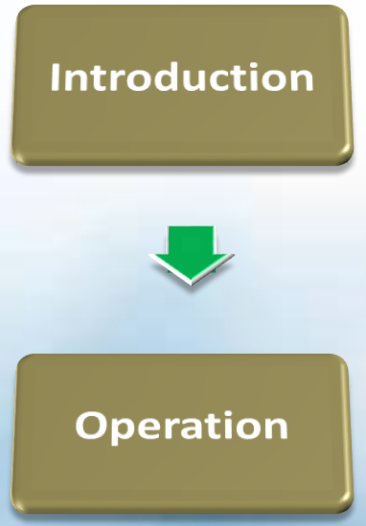
Issues in CASE Life Cycle – 2. Tailoring



- The activities required to customize a CASE system for a particular organization and application domain include:-
  - ***Installation*** – installed & tested on org's h/w
  - ***Process model definition*** – allows CASE mgr to see where the tool can be applied & what interfaces to other tools may need to be constructed
  - ***Tool integration*** – develop interface modules to integrate different tools
  - ***Documentation***

Issues in CASE Life Cycle





1. User Resistance
2. Lack of Training
3. Management Resistance