

Key Topics

- › Microsoft Project
- › COCOMO
- › Planview
- › DOORS
- › Rational Software Modeller
- › LDRA Testbed
- › Integrated Development Environment
- › Quality Center

8.1 Introduction

The goal of this chapter is to give a flavour of a selection of tools¹ that can support the organization in various software engineering activities. The tools considered are for project management, configuration management, design and development, testing, and so on. The organization will generally choose tools to support the process rather than choosing a process to support the tool.²

Mature organizations will employ a structured approach to the introduction of new tools. First, the requirements for a new tool are specified and the available options to

¹The list of tools discussed in this chapter is intended to give a flavour of what tools are available, and the inclusion of a particular tool is not intended as a recommendation of that tool. Similarly, the omission of a particular tool should not be interpreted as disapproval of that tool.

²That is, the process comes first and then the tool rather than the other way around.

satisfy the requirements are considered. These options may include developing a tool internally; outsourcing the development of a tool to a third party supplier; or purchasing a tool off-the-shelf from a vendor.

Several candidate tools will be considered prior to selection, and each candidate tool will be evaluated to determine the extent to which it satisfies the specified requirements. An informed decision is then made and the proposed tool will be piloted in the organization prior to its deployment. The pilot provides feedback on its suitability and will be considered prior to a decision on full deployment and whether any customization is required prior to rollout.

Finally, the users are trained on the tool, and the tool is rolled out throughout the organization. Support is provided as required for a period post-deployment. The first area for which we consider tools is project management.

8.2 Tools for Project Management

There are several tools to support project management activities such as estimation and cost prediction, planning and scheduling, monitoring risks and issues, and managing a portfolio of projects. These include tools such as Microsoft Project which is a powerful project management scheduling tool widely used by project managers throughout the world. Small projects may employ a simpler tool such as Microsoft Excel for their project scheduling activities.

The Constructive Cost Model (Cocomo) is a cost prediction model originally developed by Barry Boehm [Boe:81] to estimate effort, schedule, and cost for small and medium projects. It is based on an effort estimation equation that calculates the software development effort in person-months from the estimated project size. The estimated effort for requirements and maintenance is then derived from this.

The effort estimation calculation is based on the estimate of a project's size in thousands of source lines of code (SLOC³). The accuracy of the tool is limited as there is a great deal of variation among teams due to differences in the expertise and experience of the personnel in the project team.

There are several commercial variants of the tool including the Cocomo basic, intermediate, and advanced models. The intermediate model includes several cost drivers to model the project environment, and each cost driver is rated. There are over 15 cost drivers used and these include product complexity, reliability, and experience of personnel as well as programming language experience. The Cocomo parameters need to be calibrated to reflect the actual project development environment. The effort equation used in Cocomo is given by

$$\text{Effort} = 2.94 * \text{EAF} * (\text{KSLOC})^E \quad (8.1)$$

³SLOC includes delivered source lines of code created by project staff (excluding automated code generated and also code comments).

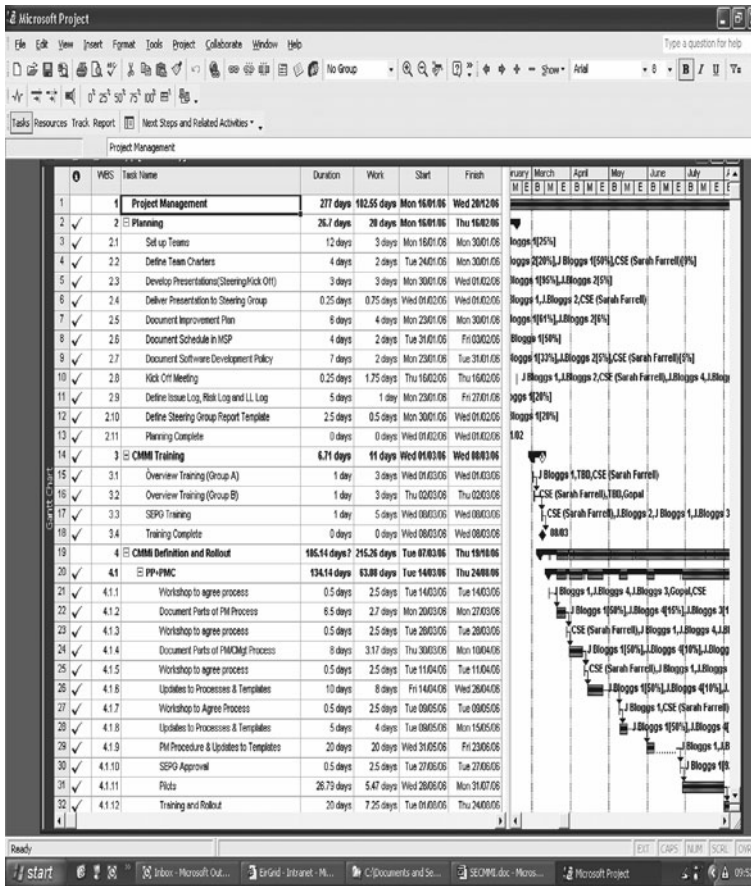


Fig. 8.1 Microsoft Project schedule

In this equation, EAF refers to the effort adjustment factor that is derived from the cost drivers, and E is the exponent that is derived from the five scale drivers.⁴ Costar tool is a commercial tool that implements the Cocomo model and may be used on small or large projects. It needs to be calibrated to reflect the particular software engineering environment, and this will enable more accurate estimates to be produced.

Microsoft Project is a project management tool that is used for planning, scheduling, and charting project information. It enables a realistic project plan to be created, and the plan will be revised regularly during the project to reflect actual progress, and the project is re-planned as appropriate (Fig. 8.1).

⁴The five scale drivers are factors contributing to duration and cost and they determine the exponent used in the Effort equation. Examples include team cohesion and process maturity.

A project is defined as a series of steps or tasks to achieve a specific goal. The amount of time that it takes to complete a task is termed its duration and tasks are performed in a sequence determined by the nature of the project. Resources such as people and equipment are required to perform a task. A project will typically consist of several phases such as creating the plan, tracking and managing the project, and closing the project.

The project schedule details the tasks and activities to be carried out during the project; the effort and duration of each task and activity; the percentage completed of each task, and the resources needed to carry out the various tasks. The schedule shows how the project will be delivered within the key project parameters such as time, cost, and functionality without compromising quality in any way.

The project manager is responsible for managing the schedule and will take corrective action when project performance deviates from expectations. The project schedule will be updated regularly to reflect actual progress made and the project re-planned appropriately.

Project portfolio management (PPM) treats a set of projects as a portfolio and is an effective way to manage a group of current or proposed projects. It allows the organization to choose the optimal mix and sequencing of its projects to yield the greatest business benefit to the organization.

Table 8.1 Key capabilities of Planview Enterprise

Capability	Description
Strategic planning	Define mission, objectives, and strategies Allocate funding/staffing for chosen strategy Automate and manage strategic process
Investment analysis	Devise strategic long-term plans Identify key criteria to evaluate initiatives Optimize strategic and project investments to maximize business benefit
Capacity management	Balance resources with business demands Ensure capacity supports business strategy Align top-down and bottom-up planning Forecast resource capacity
Demand management	Request work and check status Review life cycles
Project management	Scope, schedule, and execution of work Track/report time worked against projects Track and manage risks and issues Track/display performance and trend analysis
Financial management	Collaborate to better forecast cost Monitor spending
Resource management	Balance portfolios/assign people efficiently Improve forecasting Keep staff productive
Change management	Determine impact of change on schedule/cost Effectively manage change

It analyses the project’s total expected cost, the resources required, the schedule, the benefits that will be realized as well as interdependencies with other projects in the portfolio. This allows project investment decisions to be made methodically to deliver the greatest benefit to the organization. This is a shift from the normal once off analysis of an individual project proposal, to the analysis of a portfolio of projects. PPM tools aim to manage the continuous flow of projects from concept all the way to completion.

There are several commercial portfolio management tools available from various vendors. These include Clarity PPM from Computer Associates, ChangePoint from Compuware, RPM from IBM Rational, PPM Center from HP, and Planview Enterprise from Planview. It is not possible to discuss all of these portfolio management tools, and the discussion in this section is limited to the Planview Enterprise tool.

Planview allows organizations to manage projects and resources across the enterprise and to align their initiatives for maximum business benefit. It provides visibility into and control of strategic and project portfolios and allows the organization to prioritize and manage its projects and resources. This allows it to make better investment decisions and to balance its business strategy against its available resources. Planview Enterprise Portfolio Management helps an organization to optimize its business through eight key capabilities (Table 8.1).

Planview provides dashboard views that allow key project performance indicators to be closely tracked. This includes dashboard views of variances of cost, effort, and schedule as described in Fig. 8.2.

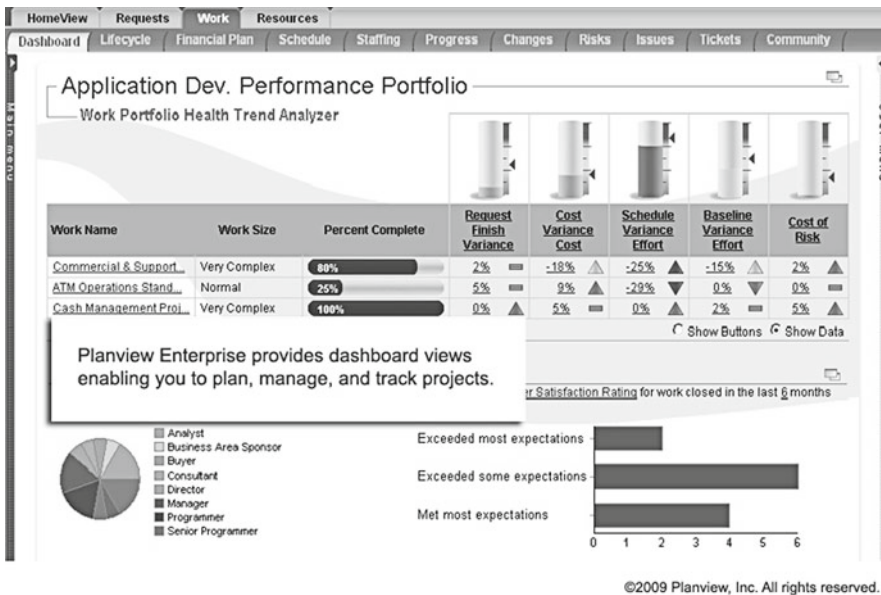
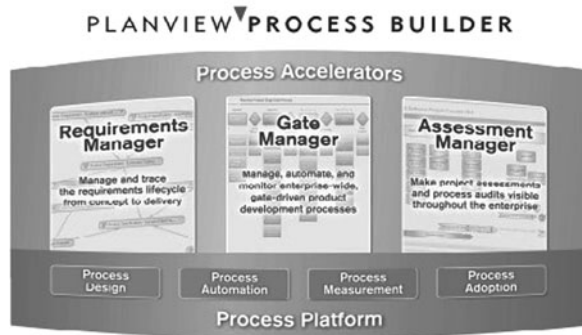


Fig. 8.2 Dashboard views in Planview Enterprise (Courtesy of Planview)

Fig. 8.3 Planview Process Builder (Courtesy of Planview)



Planview includes Process Builder (Fig. 8.3), which allows modelling and management of enterprise-wide processes. It provides improved tracking, control, and audit capabilities in key process areas such as requirements management and product development, as well as satisfying key regulatory requirements.

The organization may define and model its process in Process Builder, and this includes process adoption, compliance, and continuous improvement. The functionality includes

- Process design
- Process automation
- Process measurement
- Process auditing

Next, we will consider tools to support requirements development and management.

8.3 Tools for Requirements

There are many tools available to assist organizations in carrying out requirements development and management. These tools assist in eliciting requirements from the stakeholders; modelling requirements; verifying and validating the requirements; managing the requirements throughout the life cycle; and providing traceability of the requirements to the design and test cases. The following is a small selection of some of the tools that are available (Table 8.2).

DOORS[®] (Dynamic Object-Oriented Requirements System) is a requirements management tool developed by IBM Rational. It allows the stakeholders to actively participate in the requirements process and aims to optimize requirements communication, collaboration, and verification. High-quality requirements help the organization in reducing costs⁵ and in meeting their business objectives.

⁵A good requirements process will enable high-quality requirements to be produced, and the cost of poor quality is reduced as wastage and re-work is minimized. The requirements are the foundation

Table 8.2 Tools for requirements development and management

Tool	Description
DOORS (IBM Rational)	This is a requirements management tool developed by Telelogic (which is part of IBM Rational)
Requisite Pro (IBM Rational)	This is a requirements management and use case management tool developed by IBM Rational
Enterprise Architect (Sparx Systems)	This is a UML analysis and design tool that covers requirements gathering, through to analysis and design, to testing and maintenance. It is developed by Sparx Systems and integrates requirements management with the other software development activities
CORE (Vitech)	This is a requirements tool developed by Vitech and may be used for modelling and simulation
MKS integrity	This tool was developed by MKS and enables organizations to capture and validate software requirements and to link them to downstream development and testing activities

The tool can capture, link, trace, analyse, and manage changes to the requirements. It enhances communication and collaboration to ensure that the project conforms to the customer requirements, as well as compliance to regulations and standards.

Requirements are documented in a way that is easy to interpret and navigate. It is easy to locate information within the database, and the user requirements are recorded in a document style showing each individual requirement. It provides views of the list with assigned identifiers and also an Explorer-like navigation tree.

The DOORS tool employs links to support traceability of the requirements. Links are traversed with a simple click of the mouse to the corresponding object. The links are easy to create by dragging and dropping, e.g. a new link from the user requirements to the system requirements is created in this way. The tool provides dynamic reporting on traceability and filters may be employed to ensure that traceability is complete. Traceability is essential in demonstrating conformance to the requirements.

The management of change is an important part of the requirements process. The DOORS tool supports changes to requirements and allows an impact analysis of the proposed changes to be performed. It allows changes that could impact other requirements or design items and test cases to be tagged (Fig. 8.4).

The IBM Rational DOORS[®] tool

- Provides a comprehensive requirements management environment
- Provides web browser access to the requirements database
- Manages changes to requirements
- Provides scaleable solution for managing project scope and cost

of the system and if they are incorrect then irrespective of how good design and development are the delivered system will not be fit for purpose.

IBM Rational Doors

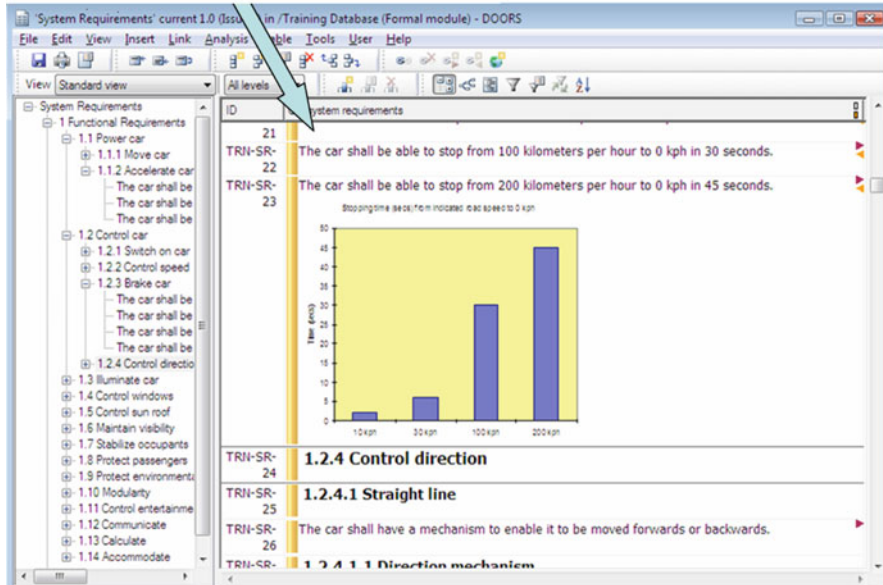


Fig. 8.4 IBM Rational DOORS tool (Courtesy of IBM)

- Provides traceability to design items, test plans, and test cases
- Provides active engagement from stakeholders
- Integrates with other IBM Rational tools

There are several other IBM Rational tools that may be integrated with DOORS[®]. These include System Architect, Requirements Composer, Rhapsody, and Quality Manager.

IBM Rational RequisitePro is a requirements management tool that allows requirements to be documented with familiar document-based methods as well as providing database-enabled capabilities such as requirements traceability and impact analysis. Requirements may be managed throughout the life cycle and changes to the requirements controlled.

The CORE product suite is developed by Vitech and has functionality for requirements management, modelling and simulation, and verification and validation. It supports UML activity and sequence diagrams, which enables the desired behaviour and flow of control to be captured and analysed. The tool provides

- Comprehensive end-to-end system traceability
- Change impact analysis
- Multiple modelling notations with integrated graphical views
- System simulation based on behavioural models
- Generation of documentation from the database

The Integrity tool was developed by MKS and it enables organizations to capture and validate software requirements. It enables them to link the requirements to downstream

development and testing activities and to manage changes to the requirements. Next, we will consider tools to support software design and development

8.4 Tools for Design and Development

Once the system and software requirements have been specified the designers are in a position to devise an appropriate technical solution to satisfy them. This section describes tools available to support design and software development activities. The software design includes the high-level architecture of the system, as well as the lower level design and algorithms. There are various tools to support designers (Table 8.3).

IBM Rational Software Modeler[®] (RSM) is a UML-based visual modelling and design tool. It promotes communication and collaboration during design and development and allows information about development projects to be specified and communicated from several perspectives. It is used for model-driven development and aligns the business needs with the product.

It gives the organization control over the evolving architecture and provides an integrated analysis and design platform. Abstract UML specifications may be built with traceability and impact analysis shown.

It has an intuitive user interface and a diagram editor to create expressive and interactive diagrams. The tool may be integrated with other IBM Rational tools such as Clearcase, Clearquest, and Requisite Pro (Fig. 8.5).

Table 8.3 Tools for software design

Tool	Description
Microsoft Visio	This tool allows many types of drawings such as flow charts, work flow diagrams, and network diagrams to be created
IBM Rational Software Modeler	This is a UML-based visual modelling and software design tool
IBM Rational Rhapsody	This modelling environment tool is based on UML and provides a visual development environment for software engineers. It uses graphical models and generates code in C, C++, and Java
IBM Rational Software Architect	This modelling and development tool uses UML for designing architecture for C++ and Java applications
Enterprise Architect (Sparx Systems)	This UML analysis and design tool is used for modelling business and IT systems with traceability from requirements to analysis and design, testing and deployment. It supports code generation

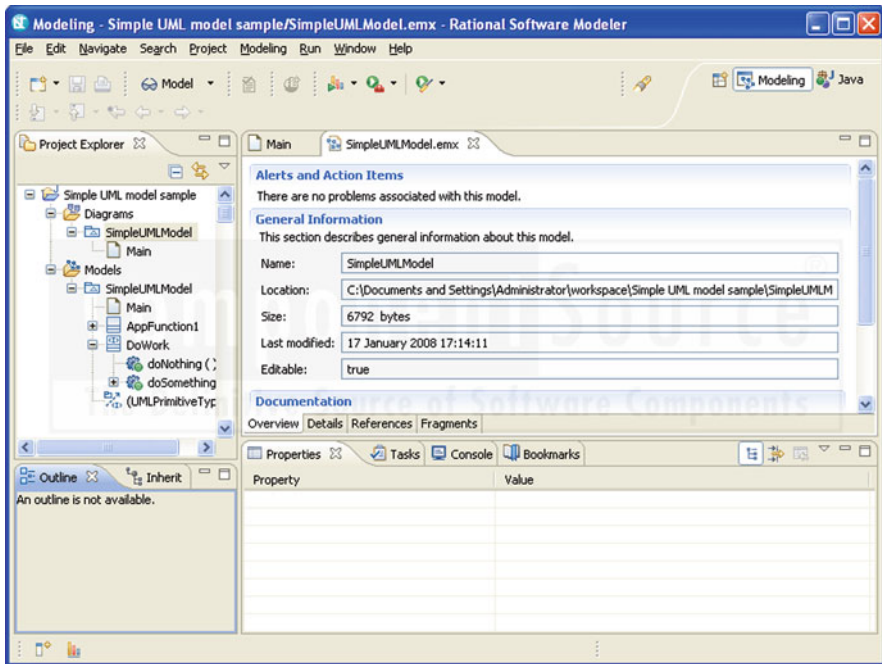


Fig. 8.5 IBM Rational Software Modeler (Courtesy of IBM)

BM[®] Rational[®] Rhapsody[®] is a visual development environment used in real-time or embedded systems. It helps teams collaborate to understand and elaborate requirements; abstract complexity using modelling languages such as UML; validate functionality early in development; and automate code generation to speed up the development process.

Enterprise Architect is developed by Sparx Systems and is a UML analysis and design tool used for modelling business and IT systems. It covers the full product development life cycle, including business modelling, requirements management, software design, code generation, testing. It supports automated document generation, code generation, and reverse engineering of source code. Its reverse engineering feature allows a visual representation of the software application to be provided (Fig. 8.6).

It is a multi-user graphical tool with built-in reporting and documentation. It can model, manage, and trace requirements to design, testing, and deployment, and it can trace the implementation of system requirements to model elements. It can search and report on requirements and perform an impact analysis on proposed changes to the requirements.

The tool allows deployment scripts to be built, debugged, tested, and executed from within its development environment. UML and modelling are integrated into the development process and debugging capabilities are provided. This includes run-time examination of the executing code for several programming languages, and NUnit and JUnit test classes may be generated and integrated directly into the test process.

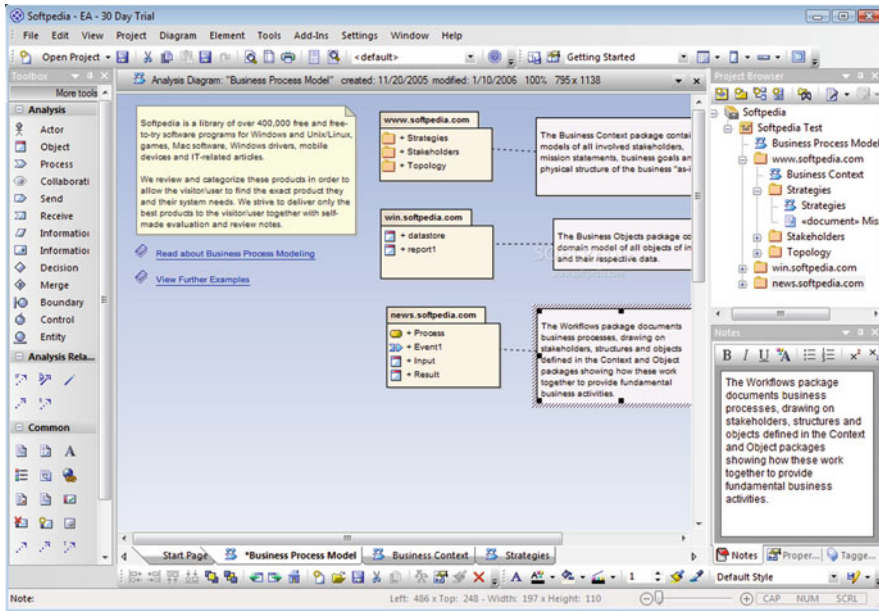


Fig. 8.6 Sparx Enterprise Architect (Courtesy of Sparx Systems)

An integrated development environment (IDE) is a software application that provides comprehensive support facilities to software developers. It includes specialized text editors; a compiler; build automation; and debugging capabilities. The features of an IDE are described in Table 8.4.

IDEs help to improve programmer productivity. They are usually dedicated to a specific programming language, although there are some multi-language tools such as Eclipse and the latest versions of Microsoft Visual Studio.

There are many IDEs available such as Microsoft Visual Studio for C and C++, JBuilder and JCreator for Java, Free Pascal for Pascal. The next section is concerned with tools to support configuration management.

8.5 Tools for Configuration Management and Change Control

Configuration management is concerned with identifying the work products that are subject to change control and controlling changes to and releases of these work products. It involves creating baselines and maintaining the integrity of baselines by providing accurate configuration data to stakeholders. It records and reports the status of configuration items and change requests and verifies the correctness and completeness of configuration items with configuration audits.

Table 8.4 Integrated development environment

Item	Description
Source code editor	This is a specialized text editor designed for editing the source code. It includes features to simplify and speed up the input of source code, including syntax checking of the code while the programmer types. Examples include Microsoft Visual Studio
Compiler or interpreter	A compiler is a computer program that translates the high-level programming language source code into object code to produce the executable code. A compiler carries out lexical analysis, parsing, and code generation An interpreter is a program that executes instructions written in a programming language. It may involve the direct execution of the code; translation of the code into an intermediate representation and immediate direct execution; or execution of stored precompiled code made by a compiler which is part of the Interpreter System
Build automation tools	Build automation involves scripting to automate the build process. This includes tasks such as compiling the source code; linking the object code and building the executable software; performing automated tests and reporting results; reporting the build status; and generating release notes
Debugger	A debugger is a software application that is used to debug and test other software programs. Debuggers offer step-by-step execution of the code or execution to breakpoints in the code. Examples include IBM Rational Purify and Microsoft Visual Studio Debugger

Visual Source Safe (VSS) is a version control management system for source code and binary files. It is developed by the Microsoft Corporation and is used mainly by small software development organizations. It allows multiple users to place their source code and work products under version control management. It is fairly easy to use and may be integrated with the Microsoft Visual Studio tool. Microsoft plans to replace VSS with its Team Systems tool.

Polytron Version Control System (PVCS) is a version control system for software code and binary files. It is developed by Serena Software Inc. and is suitable for use by large or small teams. It allows multiple users to place their source code and project deliverables under version control management, and it allows files to be checked in and checked out; baselines to be controlled; rollback of code; and tracking of check-ins. It includes functionality for branching, merging, and labelling. It includes the PV Tracker tool for tracking defects and the PV Builder tool for performing builds and releases.

The PV Tracker tool automates the capture and communication of issues and change requests. This is done throughout the software development life cycle for project teams, and the tool allows the developers to link the affected source code files with issues and changes. It allows managers to determine and report on team progress and to prioritize tasks. PV Builder maintains an audit trail of the files included in the build as well as their versions.

The IBM® Rational® Clearcase and Clearquest tools have a rich feature set, and they are among the most popular configuration management tools. Clearcase allows software code and other software deliverables to be placed under version control management, and it may be employed in large or medium projects. It can handle a large number of files and supports standard configuration management tasks such as checking in and checking out of the software assets as well as labelling and branching. Objects are stored in repositories called VOBs.

Clearquest may be linked to Clearcase as well as to other IBM Rational tools. It allows the defects in a project to be tracked, and it allows the versions of source code modules that were changed to be linked to a defect number in Clearquest.

8.6 Tools for Code Analysis and Code Inspections

Static code analysis is the analysis of software code without executing the code. It is usually performed with automated tools and the analysis conducted depends on the sophistication of the tools. Some tools may analyse individual statements or declarations, whereas others may analyse the whole source code. The objective of the analysis is to highlight potential coding errors early in the development life cycle.

The LDRA Testbed Tool automatically determines the complexity of the source code, and it provides metrics that give an indication of the maintainability of the code. A useful feature of LDRA is that it gives a visual picture of system complexity, and it has a refactoring tool to assist with reducing complexity. It automatically generates code assessment reports listing all of the files examined and providing metrics of the clarity, maintainability, and testability of the code (Fig. 8.7).

Compliance to coding standards is important in producing readable code and in preventing error-prone coding styles. There are several tools available to check conformance to coding standards including the LDRA TBvision tool which has reporting capabilities to show code quality as well as fault detection and avoidance measures. It provides users with functionality to view the results presented intuitively in various graphs and reports.

Some static code analysis tools (e.g. tools for formal methods) aim to prove properties about a particular program. This may include reasoning about program correctness or that of a program meeting its specification. These tools often provide support for assertions, and a precondition is the assertion placed before the code fragment, and this predicate is true before execution of the code. The post-condition is the assertion placed after the code fragment, and this predicate is true after the execution of the code.

There are several open-source tools available for static code analysis, and these include the RATS tools which provide multi-language support for C, C++, Perl, and PHP and the PMD tool for Java. There are several commercial tools available, and these include those that provide multi-programming language support and those that support a specific language. The LDRA Testbed tool provides support for C, C++, and Java; the Fortify tool helps developers to identify security vulnerabilities in C, C++, and Java; and the Parasoft tool helps developers to identify coding issues that lead to security, reliability, performance, and maintainability issues later.

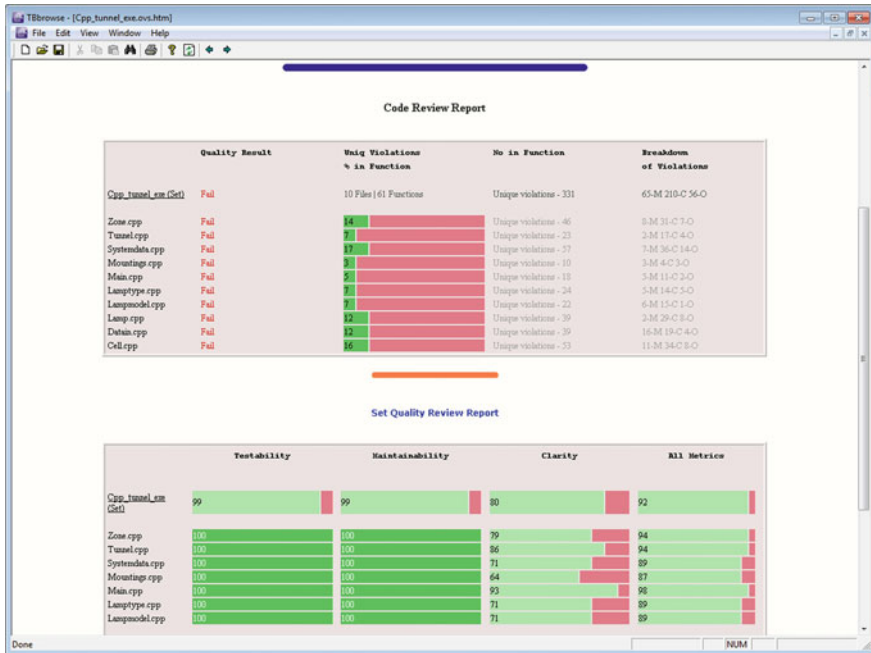


Fig. 8.7 LDRA Testbed – quality review report (Courtesy of LDRA Systems)

8.7 Tools for Testing

Testing plays a key role in verifying that the software system satisfies the requirements and is fit for purpose. There are various tools to support testing such as test planning and management tools; defect tracking tools; regression test automation tools; performance tools. The tools considered in this section include

- Test Director and Quality Center
- Winrunner
- Load Runner

Test Director is a web-based test management tool developed by Mercury⁶ It provides a consistent repeatable process for gathering requirements; planning and scheduling tests; analysing results; and managing defects. It consists of four modules namely

⁶Mercury is now part of HP.

- Requirements
- Test Plan
- Test Lab
- Defect Management

The Requirements module supports requirements management and traceability of test cases to the requirements. The Test Plan module supports the creation and update of test cases. The Test Lab module supports execution of the test cases defined in the Test Plan module. The Defect Management module supports the logging of defects and these defects can be linked back to the test cases that failed.

Test Director supports a high level of collaboration and communication between the stakeholders. It allows the business analysts to define the application requirements and testing objectives. The test managers and testers may then design test plans, test cases, and automated scripts. The testers then run the manual and automated tests, report results, and log the defects. The developers review and correct the logged defects. Project and test managers can create status reports and manage test resources. Test and product managers decide objectively whether the application is ready to be released.

The Test Director tool is now a part of Quality Center™ (Fig. 8.8) developed by HP. This tool standardizes and manages the entire test and quality process and is a web-based system for automated software quality management and testing. It employs dashboard technology to give visibility into the process.

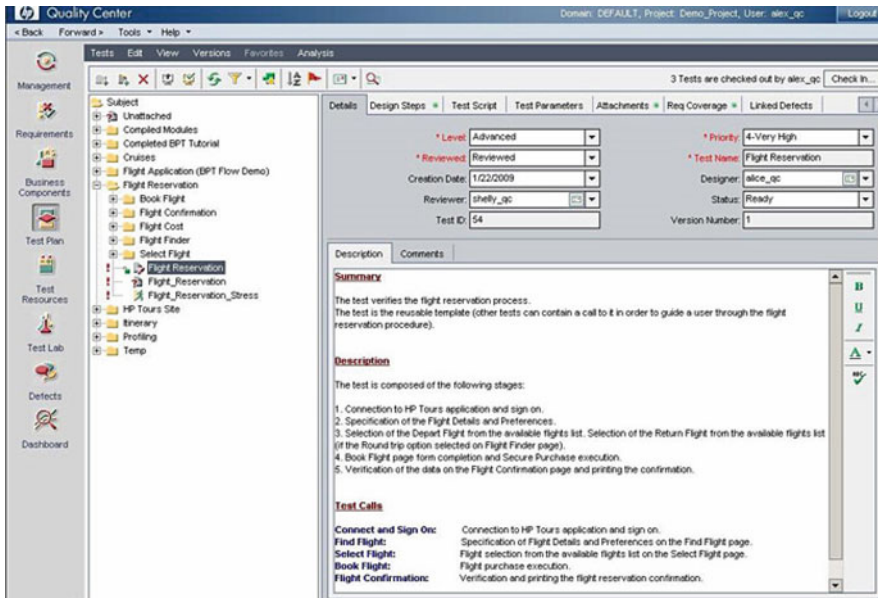


Fig. 8.8 HP Quality Center (Courtesy of HP)

Mercury developed the Winrunner tool which automatically captures, verifies, and replays user interactions. It enables defects in applications to be easily identified, and it is mainly used to automate regression testing. This helps to improve test quality productivity and allows regression testing to be performed in a timely manner. This provides confidence that enhancements to the software have had no negative impact on the integrity of the system. The Winrunner tool has been replaced by HP Unified Functional Testing software which includes HP Quick Test Professional and HP Service Test.

Mercury developed the LoadRunner testing tool, and this tool allows the software application to be tested with hundreds or thousands of concurrent users to determine its performance under heavy loads. It allows the scalability of the software system to be determined to check if its performance can support future predicted growth.

8.8

Review Questions

1. Why are tools used in software engineering?
2. How should a tool be identified for an organization and how should it be deployed?
3. What is the relationship between the process and the tool? Which comes first?
4. What tools would you recommend for project management?
5. Describe how you would go about selecting a tool for requirements development.
6. Describe various tools that are available for design and development and which are your preferred tools?

8.9

Summary

The objective of this chapter was to give a flavour of various tools available to support the organization in engineering software. These included tools for project management, configuration management, design and development, test management. The tools are generally chosen after the process has been defined, i.e. the process comes first and then the tools.

The focus on tools for project management was on tools to support project estimation and scheduling and portfolio management. The discussion included the Cocomo cost model which may be employed to estimate the cost and effort for a project and the Microsoft Project tool which is used extensively by project managers to schedule and track their projects. The Planview Portfolio Management Tool was also discussed and this tool allows an organization to choose the optimal mix of projects to maximize its return on investment.

The discussion on tools to support requirements development and management included tools such as IBM Rational DOORS, Requisite Pro, and CORE. The DOORS tool allows all stakeholders to actively participate in the requirements process and aims to optimize requirements communication, collaboration, and verification throughout the organization.

The discussion on tools to support design on development included the IBM Rational Software Modeler tool, the Sparx Enterprise Architect tool, and Integrated Developer Environments to support software developers. The Rational Software Modeler[®] (RSM) is a UML-based visual modelling and design tool. Enterprise Architect is a UML analysis and design tool used for modelling business and IT systems. It provides traceability from requirements to analysis and design, testing and deployment, and supports code generation. The discussion on tools to support configuration management included PVCS and Clearcase.

The discussion on tools to support testing included the Test Director, Winrunner, and Loadrunner tools. Test Director is a test management tool with modules for requirements management, test planning, test lab, and defect management. Test Director is now a part of Quality Center[™], and this tool standardizes and manages the entire test and quality process.

Tool selection is done in a controlled manner. First, the organization needs to determine its requirements for the tool. Next, it will need to decide on whether an in-house solution will be developed or whether to purchase a tool from a vendor. Various candidate tools are evaluated and a decision on the proposed tool is made. Next, the tool is piloted to ensure that it meets the needs of the organization, and feedback from the pilot may lead to changes or customizations of the tool. Finally, the end users are trained on the use of the tool and it is rolled out throughout the organization.