

# Creating a Metrics Program

## Step 6: Assemble a Metrics Toolset

Step 6 is to identify and configure tools which are necessary to support the metrics program.

To have an effective metrics program, you must put tools in place that facilitate ease of data collection, and effective process support and project management. As well, tools chosen should integrate smoothly with other existing tools (such as those for analysis, storage or presentation). Assess what is already available to you. You may need to acquire tools commercially or develop them in-house.

### A Different Tool for Every Job

Figure 3.22 links the data groups introduced in Step 5 with the types of tools best suited to collecting the particular data. Each type of tool in Figure 3.22 is described in the following section, and examples of real tools are given. It should be noted that the lists of tools presented here are not exhaustive, nor are any recommendations implied.

### So Many Tools, So Much Data

Each type of tool has its own advantages. In selecting tools, consider general purpose tools designed for collecting many types of data versus special purpose tools that are more efficient at certain tasks.

Figure 3.22 :  
Data groups by  
tool type.

Data Group	Types of Tools
estimation	spreadsheet, cost estimating, project scheduling
effort	time sheets, project status reports
staffing	project scheduling
project management	project scheduling, configuration management, project status reports
financial	spreadsheet, project scheduling, accounting
requirements	requirements trace, CASE, configuration management, defect management
implementation	code analysis, configuration management, project status reports
testing	test management
defect	defect management, spreadsheet, database
performance	performance monitor

## **CASE Tools**

Computer-aided software engineering (CASE) tools generally support requirements analysis and design activities. You can use them to determine the number of requirements, and possibly to track the number of requirements added, changed or deleted (through their trace features or change control features).

Examples tools include Teamwork from Cadre Technologies, Software Through Pictures/Object Modeling Tool from Interactive Development Environments, TurboCase, Structured Architect from Popkin Software & Systems Inc., POSE, Excelerator II, Open Select, Developer, and Rose (PC/Rational computer environments).

## **Requirements Trace Tools**

Requirements trace tools enumerate requirements based on textual documents or on CASE tools. You can use them to count requirements and to monitor the evolution of requirements. One example is TWK/RTRACE from Cadre Technologies.

## **Configuration Management Tools**

Configuration management (CM) tools support both version control and change control. Configuration management tools generally produce lists of all the software units controlled by the tool and their status (such as inspected or completed). Change control aspects of CM tools support the enumeration of requirements additions, changes and deletions. Examples include CCC/Manager from Softool Corp., PVCS from Intersolve, SourceSafe from One Tree, CMCV/6000 for IBM, DDTS from Qualtrack, and ProTEAM from Scopus.

## **Database Tools**

You can use database tools to quickly set up forms and reports for recording and reporting defect data. Examples include ACCESS, FoxPro, dBASE IV, and Ingres.

## **Defect Management Tools**

Defect management tools are specific to problem and fault management tasks. Some may also have the ability to track software action items (SAIs) originating in review meetings. You may use these tools to record requirements changes, as well as data about defects (both problems reported and faults discovered in testing). Examples include Defect Control from Software Edge, and many of the Configuration Management tools.

## **Code Analysis Tools**

Code analysis tools are available to count lines of source code, lines of documentation, and to analyze complexity. They are typically specific to the language, platform and (sometimes) compiler, and are often developed in-house. Tools for extracting information from code include programs for counting code and documentation, and complexity analysis tools.

Code counting programs are typically small programs that accept a set of code files as input and count the number of non-blank, non-commented lines and the number of commented lines (LOD). The definition of a valid line of code will vary between companies. You must choose an algorithm that is familiar to your staff and above all, consistent.

Complexity analysis tools scan source code and estimate the complexity of the software. Examples include the tool CHECKPOINT by Software Productivity Research, UX-METRIC from Set Laboratories, and McCabe Tools from McCabe and Associates.

## **Test Management Tools**

Test management tools support the enumeration, status and execution of tests and regression testing. Some will also determine code coverage of a suite of tests. Code coverage tools are typically built in or added on to automated test case execution software. These tools identify which program branches or statements are executed by a set of test cases. You can use them in a metrics program to collect testing data. Examples include Test Manager, TestPro, TestPro for Windows, Memcheck, Dr. Taylor's Test, Test, and McCabe Tools from McCabe and Associates.

## **Performance Monitoring Tools**

Performance monitoring tools analyze the use of computer resources by the operational software. They determine how busy the CPU is, how much memory is being used, and how busy the communications systems are. These tools help to isolate resource intensive software units. Examples include perfmeter from Sun, the ps command in the Unix environment, and MSD from MicroSoft.

## **Cost-Estimating Tools**

Cost-estimating tools use estimates of software size in SLOC or function points as input along with a number of parameters that influence productivity. They apply software cost models (such as COCOMO) to estimate software effort, cost, labor rate, staffing profiles and schedules for each software activity. Examples include ESTIMATE Professional, JS-3, SLIM, PCOC, WICOMO, PRICE-S, SoftCost, SPQR/20.

## **Time Sheet Tools**

Time sheet tools support the recording and reporting of effort (person hours) per activity. They are an effective way to collect effort data per activity if you accurately code and record actual time spent. Examples include TimeSheet Professional for Windows from Timeslips Corporation, and Timex shareware.

## **Accounting Tools**

Accounting tools focus on general ledger and other accounting tasks but you may use them to report total budgets and budget allocated to support staff and tools. Examples include Peachtree, Quickbooks, One Write Plus, CA Simply Accounting, and DacEasy Instant Accounting.

## **Project Scheduling Tools**

Project scheduling tools support the collection of data concerning schedule, effort and cost. They often interface directly with spreadsheet tools for data interchange. Over 100 of these tools exist. Some examples are CA-SuperProject, Harvard Project Manager, MacProject II, MicroPlanner, MicroSoft Project, Milestones Etc., OnTarget, Project Outlook, Project Scheduler 5, Time Line, and PrimaVera.

## **Project Status Reporting Tools**

Data reported initially in project status reports includes the number of (overtime) hours worked, actual start/completion dates of each activity, number of units required/completed in each activity, and number of units inspected. Tools range from simple text editors using templates to sophisticated forms packages. Most editors, word processors, or database software can produce project status reports. Being able to email the report can be an asset, if incorporated into the tool.

## **Spreadsheets**

Spreadsheets tally data in simple rows and columns format. You can use them when collecting estimation and financial data. You can also use spreadsheets effectively for defect reporting, including software action item reporting. Examples include Excel, Lotus 123, Wingz and Quattro Pro.

---

## **Actions Required for Step 6**

1. Consult Step 5 to determine in which groups of data your metrics belong. Use Figure 3.22 to determine which types of tools should be considered as candidates. Review the discussion of each tool type in the section to decide which tools to use to support the metrics program. Consider the impact of the tools on the collection procedures defined in Step 5.
  2. Investigate the available commercial tools if necessary to determine which are appropriate. Purchase tools as required.
  3. Design and develop any other needed tools internally.
  4. Install and set up the selected tools as best serves the development environment.
  5. Train the people who will be using the tools.
-

## **Step 6 Example**

### **Toolset**

You examine the tools required to support the data collection procedures and make several recommendations.

Buddicorp currently has a payroll database from which a portion of the metrics data will be collected. You suggest that Buddicorp implement a customized payroll database report for the metrics coordinator. In the long term, the report can evolve into an automated batch update of the metrics database.

Buddicorp has been leaving the use of automated project management and scheduling tools to the discretion of the project managers. You recommend that Buddicorp evaluate commercially available tools and select one as the company standard. The company must require that all future projects use the tool.

Since some of the metrics data must be collected when software documentation changes, you suggest that Buddicorp start maintaining project documentation in the configuration management system. It will have to create guidelines for the application of configuration management to documentation files.

Buddicorp requires code counting programs for its source code, which consists of C, Pascal, and Assembler. The code counting tool must be able to count lines of comments in the respective source files. You recommend that Buddicorp create such a tool internally. The company can use it in conjunction with the configuration management system, counting lines of code when programs are checked in.

You decide that the existing Buddicorp defect tracking system is sufficient for the initial metrics program.