Software Measurement Programs and Industry Leadership

Capers Jones
Software Productivitiy Research Inc.

Consulting studies carried out by the author and his colleagues at several hundred companies and some government agencies shows that the leading organizations tend to have very sophisticated quality and productivity measurement programs in place. This article covers the highlights of the measurement programs noted among companies in the top 25 percent of productivity and quality results in the United States.

This author and Software Productivity Research Inc. were able to gather data on software projects from more than 600 companies and about 25 government agencies. The total volume accumulated since 1985 amounts to more than 10,000 software projects. Most of our clients are large Fortune 500 corporations although we have done studies for several defense contractors, as well as for the Air Force and Navy.

This article summarizes the best measurement practices noted among these clients, which are in the upper quartile of overall productivity measured in terms of function points per staff month. Unfortunately there are no military or defense projects in the upper 25 percent of productivity data, or even in the upper half. This is due to older military standards such as the Department of Defense (DoD) 2167, which triggered the production of specifications and control documents nearly three times larger than civilian norms for the same size projects. Although many military projects are in the upper quartile in terms of software quality, the DoD is on the bottom in terms of productivity levels. Therefore most of the measurement practices described here were noted among large Fortune 500 companies. However, we have observed similar sets of measurements among top defense contractors.

In general, software is a troubled technology plagued by project failures, cost overruns, schedule overruns, and poor quality levels. Even major corporations such as Microsoft have trouble meeting published commitments, or shipping trouble-free software. But not all companies experience software disasters. Some have mastered software development and can achieve better than average results. When assessing successful software development companies, we always encounter sophisticated measurement programs.

In every industry there are significant differences between the leaders and the laggards in terms of market shares, technological sophistication, and quality and productivity levels. In the software industry one of the most significant differences is that the leaders know their quality and productivity levels because they measure them. The laggards do not measure. Therefore the laggards do not have a clue as to how good or bad they are. Consider three basic comparisons between your company and its competitors:

- Is your software quality better?
- Is your software productivity better?
- Is your company's time to market better?

If you cannot answer these questions, what do you think are your chances to compete against enterprises that do know the answers? If you do not know your software quality and productivity rates, your company and your job are at risk. Your company may also face litigation risk.

Measurement is only part of a suite of key factors leading to software excellence that includes:

- Good measurements.
- Good managers and technical staffs.
- Good development and maintenance processes.
- Complete software tool suites for managers and developers.
- Good organization structures.
- Specialized staff skills.
- On-the-job training.
- Good personnel policies.
- Good office environments.
- Good communications.

But measurement is a root technology that allows companies to make visible progress in improving the other factors. Without good measurements, progress is slow and may even turn negative.

Companies that do not measure tend to waste scarce investment dollars on approaches that consume time and energy but accomplish very little. Surprisingly, investment in good quality and productivity measurement programs has one of the best returns on investment of any known software technology.

Qualities that Industry Leaders Measure

The best way to decide what to measure is to find out what industry leaders measure, and then measure the same things. Following are the kinds of measurements we have noted at companies that have achieved or exceeded Level 3 on the Software Engineering Institute's (SEI) Capability Maturity Model®, have won Malcolm Baldrige awards, and are widely respected within the industry. These same companies were in the top quartile for all companies in terms of software quality and productivity levels.

Every leading company measures software quality. There are no exceptions. If a company does not measure quality, it is not an industry leader, and there is a good chance that software quality levels are hazardous. Quality is the most important topic of measurement. Here are the most important quality measures:

Customer Satisfaction

All leaders perform annual or semiannual customer satisfaction surveys. They also provide blank defect reports or customer complaint forms as part of the user manuals or inside software packaging. Defect reports via the Internet are also supported by industry leaders. Many leaders in the commercial software world have very active user groups and forums on information services. These groups often produce independent quality and satisfaction surveys.

® The Capability Maturity Model and CMM are registered trademarks of the Software Engineering Institute and Carnegie Mellon University.

Defect Quantities

Leaders keep accurate records of bugs or defects found in all major deliverables; they start early during requirements or design. At least five categories of defects are measured:

1. Requirements defects.
2. Design defects.
3. Code defects.
4. Documentation defects.
5. Bad fixes (secondary bugs introduced accidentally while fixing another bug).

Defect Removal

Leaders know the average and maximum efficiency of every major type of review, inspection, and test; they select optimum series of removal steps for projects of various kinds and sizes. Pretest reviews and inspections are normal among organizations with ultra-high quality, since testing alone is not efficient enough. Leaders remove from 95 percent to more than 99 percent of all defects prior to software delivery. Laggards seldom exceed 80 percent defect removal efficiency, and may drop below 50 percent.

Delivered Defects

Leaders begin to accumulate user-reported error statistics as soon as the software is delivered. Monthly reports are prepared and given to executives to show the defect trends against all products. These reports are summarized on an annual basis. Supplemental statistics such as defect reports by country, state, industry, client, etc. are also included.

Defect Severities

All industry leaders, without exception, use some type of severity scale to evaluate incoming bugs or defects reported from the field. The number of plateaus vary from one to five. In general, "Severity 1" are problems that cause the system to fail completely, then the severity scale descends in seriousness.

Reliability/Availability

Application reliability is normally measured using mean time between failures, or for new products mean time to failure. These measures are easier to gather for in-house applications than for commercial software since failure reports are indirect if external customers are involved. Availability is another aspect of reliability, i.e. the percentage of normal work periods the application can be used as intended. Availability is normally measured as a percentage of the work period during which the application is ready and can be run successfully. Anything under about 99 percent tends to generate dissatisfaction.

Service Response Time

This is a measure of how long it takes the software maintenance group to perform key activities such as acknowledge an incoming bug report, repair the bug, and get the fix back out to the user. Another key metric for commercial software is how long it takes a customer to report a bug to a live person. Being put on hold for more than two minutes is a bad sign. Best-in-class organizations are good in all of these factors and can turn around high-severity bugs in 24 hours or less.

Complexity

It has been known for many years that complex code is difficult to maintain and has higher than average defect rates. A variety of complexity analysis tools are commercially available that support standard complexity measures such as cyclomatic and essential complexity.

Test Coverage

Software testing may or may not cover every branch and pathway through applications. A variety of commercial tools are available that monitor the results of software testing, and help identify portions of applications where testing is sparse or nonexistent.

Cost of Quality

The basic cost-of-quality concept originated with Philip Crosby when he worked at ITT [1]. It was aimed at manufacturing and was not a perfect software fit. Therfore most major software shops have modified the original Crosby concepts when measuring. One significant aspect of quality measurement is to keep accurate records of the costs and resources associated with various forms of defect prevention and removal. For software these measures include the following costs:

- Software assessments.
- Quality baseline studies.
- Reviews, inspections, and testing.
- Warranty repairs and post-release maintenance.
- Quality tools; the costs of quality education.
- Your software quality assurance organization.
- User satisfaction surveys.
- Any litigation involving poor quality or customer losses attributed to poor quality.

Productivity and Schedule Measures

Measuring software schedules, effort, and costs are important areas that differentiate leaders from laggards. Many software leaders have also adopted function point metrics rather than the flawed *lines of code* metric. Almost all of the published benchmarks for software are expressed in terms of function points, so there are no viable alternatives for comparative studies. Here are the key productivity-related measures of leading software producers:

Size Measures

Industry leaders measure the sizes of major deliverables associated with software projects. Size data are kept in two ways. One method records the sizes of actual deliverables such as pages of specifications, pages of user manuals, screens, test cases, and volumes of source code. The second method normalizes the data for comparative purposes. Here function point metrics are the most common. Examples would be pages of specifications produced per function point, source code produced per function point, and test cases produced per function point.

Function point metrics were developed by IBM and put into the public domain in 1978. The rules for counting function point metrics are defined by the nonprofit International Function Point Users Group (IFPUG).[1]

Activity-Based Schedule Measures

Leading companies measure the schedules of every activity, and how those activities overlap or are carried out in parallel. Laggards, if they measure schedules at all, simply measure the gross

schedule from the rough beginning of a project to delivery, without any fine structure. Gross schedule measurements are totally inadequate for any kind of serious process improvement analysis.

Activity-Based Cost Measures

Leaders measure the effort for every activity, from requirements to maintenance. When measuring technical effort, leaders measure all activities, including requirements, design, coding, technical documentation, integration, quality assurance, etc. Leaders tend to have a rather complete chart of accounts with no serious gaps or omissions. Laggards either do not measure at all, or collect only project-level data that is inadequate for serious economic studies.

Three kinds of normalized data are typically created:

- Work hours per function point by activity and in total.
- Function points produced per staff month by activity and in total.
- Cost per function point by activity and in total.

Costs are the most subtle and difficult of the productivity-related measures because of large variances in salaries and even larger variances in burden or overhead rates.

Indirect Cost Measures

Leading companies measure costs of direct and indirect activities. Some of the indirect activities such as travel, meeting costs, moving, living, and legal expenses are so costly that they cannot be overlooked.

Monthly Milestone Reports

Leading companies are very good in monitoring progress and normally track every large and important project on a monthly basis. Monthly status reports include cost variance reports, milestone reports, and red-flag items, which are defined as situations that might delay the project or cause an overrun. Examples of red-flag items might be loss of key personnel, a sudden change in requirements, or some other major issue. These monthly reports are usually five to 10 pages. For large systems, first-line managers create the lowest level reports, and their work is summarized upwards. There are also reports on the completion of major milestones such as internal design, coding, and inspections.

Annual Software Reports

One of the surest signs that an organization has reached *best-in-class* status is when they produce an annual report on software demographics, productivity, quality, assessment results, and other key factors. These reports are usually produced on the same cycle as corporate annual reports (i.e. within 90 days from the close of the prior business year). Because software is usually one of the most expensive and labor-intensive commodities in history, it is appropriate to create an annual report for top corporate executives. The annual software reports for a Fortune 500 company are about 60 to 75 pages with sections for each major business unit, industry trends, technology trends, and quantitative and qualitative results.

Assessment Measurements

Even accurate quality and productivity data are of limited value unless they can explain why some projects are better or worse than others. The influential factors that affect the outcomes of software projects are normally collected by software assessments such as those performed by the SEI, Software Productivity Research, Howard Rubin Associates, Quantitative Software Management, or other consulting companies. In general, assessments cover the following topics:

- **Software Processes.** This deals with the activities from early requirements through deployment. Topics include how the project is designed, what quality assurance steps are used, and how configuration control is managed.
- **Software Tools.** There are more than 3,500 software development tools on the commercial market, and companies have built at least the same number of proprietary tools. It is considerably important to explore the usefulness of these available tools. Thoughtful companies identify gaps and missing features, and use this kind of data for planning improvements.
- **Software Infrastructure.** The number, size, and kinds of departments within large organizations are important topics, as are types of communication across organizational boundaries. Whether a project uses matrix or hierarchical management, and whether or not a project involves multiple cities or countries, exerts a significant impact on results.
- **Software Skills.** Large corporations can have more than 100 different occupation groups within their software domains. Some of these specialists include quality assurance, technical writing, testing, integration and configuration control, network specialists, and many more.
- **Staff and Management Training.** Software personnel, like medical doctors and attorneys, need continuing education to stay current. Leading companies tend to provide from 10 to 15 days of education per year for both technical staff members and software management. Assessments explore this topic.
- **Environment.** Physical office layout and noise levels exert a surprisingly strong influence on software results. The best-in-class organizations typically have fairly good office layouts, while laggards tend to use crowded cubicles or densely packed open offices. Recent additions to environmental measures include telecommuting factors and Web access. Some companies provide home computing facilities and portable computers, too.

Business and Corporate Measures

To this point, measurement has mainly been discussed at the level of individual software projects. There are also important measurements at the corporate level. Here are a few samples of corporate measurements noted among our leading clients:

Salary and Benefit Measures

Many companies perform annual benchmark studies of staff compensation and benefit levels. These are not software studies *per se*, but are carried out in support of the entire corporation.

Portfolio Measures

Major corporations can own from 250,000 to more than 1 million function points of software apportioned across thousands of programs and dozens of systems. Leading enterprises know their portfolio's size, their growth rate, replacement cost, quality levels, and many other factors. This information is important for mergers and acquisitions. It has also been a factor in tax litigation, such as ascertaining the value of the software assets when General Motors acquired Electronic Data Systems.

Market Share Measures

The industry and global leaders know a lot more about their markets, market shares, and competitors than the laggards. For example, industry leaders in the commercial software domain tend to know how every one of their products is selling in every country, and how well competitive products are selling globally.

Competitive Measures

Few companies lack competitors. Industry leaders know much about their competitors' products, market shares, and other important topics. A lot of this kind of information is available from various industry sources such as Dun & Bradstreet, Mead Data Central, *Fortune* and other journals, and from industry studies produced by organizations such as Auerbach, the Gartner Group, and others.

Summary and Conclusions

The software industry is struggling to overcome a very bad reputation for poor quality and long schedules. The companies that have been most successful in improving quality and shortening schedules have also been the ones with the best measurements.

The U.S. software industry is about to face major challenges from overseas vendors with markedly lower labor costs than U.S. norms. Measurement of software quality and productivity is already an important business tool. As off-shore software vendors use metrics and measurements to attract U.S. clients, good measurements may well become a business weapon.◆

References

1. Crosby, Philip B., *Quality is Free*, Mentor Book, New York, N.Y., 1979.

Note

1. Readers wanting more information about function point metrics can access the IFPUG Web site at www.IFPUG.org

About the Author

Capers Jones is chief scientist of Artemis Management Systems and director of Software Productivity Research Inc., Burlington, Mass. Jones is an international consultant on software management topics, a speaker, a seminar leader and author. He is also well known for his company's research programs into critical software issues:

- Software Quality: Survey of the State of the Art.
- Software Process Improvement: Survey of the State of the Art.
- Software Project Management: Suvey of the State of the Art.

Formerly, Jones was assistant director of programming technology at the ITT Programming Technology Center in Stratford, Conn. Prior to that, he was at IBM for 12 years. He received the IBM General Product Division's outstanding contribution award for his work in software quality and productivity improvement methods.

Software Productivity Research Inc.
6 Lincoln Knoll Drive
Burlington, Mass. 01803
Phone: 781-273-0140
Fax: 781-273-5176
E-mail: CJones@SPR.com
Internet: www.spr.com

Additional Readings

- Austin, Robert D., *Measuring and Managing Performance in Organizations*, Dorset House, New York, N.Y., 1996.
- Boehm, Barry W. *Software Engineering Economics* Prentice Hall, Englewood Cliffs, N.J., 1981.
- DeMarco, Tom, *Controlling Software Projects*, Yourdon Press (Prentice Hall), Englewood Cliffs, N.J., 1982.
- Dreger, J. Brian, *Function Point Analysis*, Prentice Hall, Englewood Cliffs, N.J., 1989.
- Grady, Robert B. and Caswell, Deborah L., *Software Metrics: Establishing a Company Wide Program*, Prentice-Hall Inc., 1987.
- Humphrey, Watts, *Managing the Software Process*, Addison-Wesley, Winthrop, Mass., 1989.
- Jones, Capers, *Applied Software Measurement*, McGraw-Hill, New York, N.Y., 1996.
- Jones, Capers, *Software Assessments, Benchmarks, and Best Practices*, Addison Wesley Longman, Boston, Mass., 2000.
- Kan, Stephen H., *Metrics and Models in Software Quality Engineering*, Addison Wesley Longman; Reading, Mass., 1995.
- Paulk, M.C., Curtis, B., Chrissis, Mary Beth, et al, *Capability Model for Software*, Software Engineering Institute, Pittsburgh, Pa., August 1991, SEI Technical Report 24.
- Putnam, Larry, *Measures for Excellence*, Prentice-Hall, Englewood Cliffs, N.J., 1992.
- Rubin, Dr. Howard, *Annual Software Benchmark Report for 1999*, Howard Rubin Associates, Pound Ridge. N.Y., 2000.
- Symons, Charles, *Software Sizing and Estimating Mk II Function Point Analysis*, John Wiley and Sons, 1992.

Letter to the Editor

Dear Editor:

It was so exciting to read through the articles in the November 2000 CROSSTALK and see my company and division mentioned in *Acquistion Reform May Resemble Madness, but the Method is Real*. It feels great to be recognized for the work we have done at Raytheon, Command and Control Division here in Fullerton, Calif. Here are some corrections to the dates listed in the article for the Raytheon assessments:

- The Raytheon Command and Control Division, Fullerton, Calif., assessment was performed in October 1998. The press release was released in January 1999.
- The Raytheon Missile Systems, Software Engineering Center, Tucson, Ariz., assessment was performed in November 1998.

I was fortunate to be able to participate in both externally led assessments. Again, thank you for the article. It made my day!

Sally Cheung
Engineering Process Group
Integrated Systems Division,
formerly Command and Controls Division,
Raytheon