Managing the Proportion of Testers to Other Developers

Cem Kaner, J.D., Ph.D.
Florida Institute of Technology

Elisabeth Hendrickson
Quality Tree Software, Inc.

Jennifer Brock
Ajilon Software Quality Partners

Acknowledgements

This presentation is partially based on a meeting of the Software Test Managers Roundtable (STMR 3) in Fall 2001. The meeting participants were:

Sue Bartlett
Fran McKain
Bret Pettichord
George Hamblen
Brian Lawrence
Jennifer Smith-Brock
Hung Quoc Nguyen
Laura Anneker
Elisabeth Hendrickson
Chris DeNardis
Jim Williams
Cem Kaner
Kathy Iberle
and Neal Reizer.

Measurement

At previous PNSQC's, I've laid out a set of characteristics of any software measure:

- The **attribute** being measured
  - The variability of the attribute and the type of scale appropriate to the attribute
- The **reading from the instrument** that measures the attribute
  - Variability of measurement and measurement scale
- The **relationship between the attribute and the instrument**. What causal model tells us how the instrument reading goes up or down as a function of the value of the attribute?
- The **side-effect potential**: How can we change the reading without correspondingly changing the attribute?
- The **purpose** and **scope** of the measurement

Staff Ratio Measurement

A staffing ratio is a (purported) measurement, so let's look at it. When we talk about the ratio of testers to programmers:

- ***Attribute:*** What do we mean? What's a tester? What's a programmer? What's a ratio?
- ***Instrument Reading:*** How have we been counting testers / programmers / ratios?

Ratios are very complicated, because we have two underlying measures and a relationship.

- ***Causal Model:*** What should drive the ratio up or down? If we actually increase (decrease) the relative workload of testers to programmers, does the ratio actually go up (down) or will it sometimes change in the wrong direction?
- ***Side Effects:*** Can we drive the ratio up or down without changing relative workloads? Is that a bad thing?
- ***Purpose***: Why do we want this ratio? Is it the best measure for our purposes?
- ***Scope***: Who sees this ratio? Is it subject to misunderstanding by people who are far away from the actual work?

A Puzzle

At STMR, we asked what were the managers' largest and smallest ratios of testers to other developers, and how these ratios felt:

- **There were very small ratios (1-to-7 and less) and very large ratios (5-to-1).**
- **Some of each worked and some of each failed.**
- **Many of us remembered successful projects with ratios lower than 1-to-1 more favorably than successful projects with larger ratios.**

Why is there such a range of successful ratios, and why would test managers be happy with relatively low ratios?

Defining the Attribute: These Ratios are Incommensurable

Staff	Counted as
4 programmers	programmers
1 development manager	programmer
1 test lead	Tester
1 black box tester	Tester
2 test automation engineers	Testers
1 buildmeister	Tester

1-to-1 ratio? But if there is a new build, how many black box testers are available to test it?

What IS this Ratio?

Staff	Counted as
1 programmer	programmer
1 toolsmith	programmer
1 buildmeister	programmer
1 development lead	programmer
1 development manager	programmer
1 test lead	Tester
4 black box testers	Testers

1-to-1 ratio? How many programmers are available to fix five testers' bug reports?

What IS this Ratio?

Staff	Counted as
5 programmers	programmers
5 on-site consultants (doing programming)	???
1 project team (10 people) under contract to deliver components.	???
1 full-time, on-staff test engineer	tester
3 technicians	???
3 temporary technicians (work for a contracting agency)	???
3 testers who work offsite in an independent test lab	???

How to count technicians and consultants? Is the ratio of testers to programmers 10-to-5, 10-to-20, 1-to-20, or something else?

Defining the Instrument: Incommensurable Ratios

- **Who to count?**
  - **Experienced people count the same as juniors?**
  - **Consultants, techs, contractors count the same as full-time employees?**
  - **Managers?**
- **What do the counted people do?**
  - **When programmers do testing, inspections or reviews, are they testing? (Do we count them as testers?) Even if they are testing their own code?**
  - **If testers help with debugging, are they programmers?**
- **When to start counting?**
- **Compare headcount or budgets?**

So, Measurement Problems

1. **The attribute is not well defined: We don't know what is a tester and what is a non-tester (for ratio purposes)**
2. **The meter gives us a ratio reading. Ratios should generally be suspect because they are subject to high variability. Additionally, it is unlikely for the same factors to be involved in the ratios on very small and very large projects.**
3. **What attribute does this ratio tie back to?**
4. **Now, let's consider the meaning of these ratios. Are larger ones good? Smaller ones? Specific values (like 1:1)?**

Small Ratios can be Better

- **Code coming into testing is clean, designed for testability, and has good debug support.**
- **Prevention is emphasized, and a person who makes a bug is expected to fix it.**
- **Bug churn rate is low.**
- **Staff turnover in testing and programming groups is low.**
- **Company hires skilled, experienced testers not "bodies."**
- **Shared agreement on the role of the test group.**
- **Trust and respect between programmers and testers.**
- **The groups help each other become more productive (e.g, helping them build tools).**
- **Extensive unit test library that programmers run when they update the product. (Read about Extreme Programming.)**

Small Ratios can be Worse

- **Time to market seen as more important than finding / fixing defects.**
- **Dominant market position allows seller to ship defects and charge extra for maintenance and support.**
- **Testers perceived as not contributing because they don't write code.**
- **Testers perceived as too expensive, testing is easy anyway.**
- **Testers perceived as incompetent, counterproductive twits.**
- **Test manager perceived as a whiner who uses his staff ineffectively.**
- **Test group's work perceived as poor, overemphasizing unimportant issues, or as politically motivated overemphasizing process.**
- **Toxic relation between testers and programmers, resulting in bug churn, excessive turnover.**
- **Product is so complex that it is too expensive to train new testers.**

Large Ratios can be Better

- **Product might be knitted together from many externally written components or it might be an upgrade of an existing product.**
- **Extensive configuration testing needed.**
- **Extensive documentation and repetitive labor needed because of high litigation risk (e.g. safety-critical).**
- **Extensive documentation needed for software sold in its entirety to a customer who assumes responsibility for future maintenance, support and enhancement.**
- **Market is picky about fit and finish.**
- **Load testing is needed.**
- **Testers serve multiple roles, such as domain expert, build support, archivist, network administrator, debugging, spec writer, code reviewer, benchmark competing products, etc.**

Large Ratios can be Worse

- **Testers don't understand domain or combinatorial testing, try too many redundant tests.**
- **Large numbers of low-skill testers.**
- **Manual execution of large sets of fully scripted test cases.**
- **High test group turnover, constantly in training mode.**
- **Testers have inadequate tools, space, and equipment.**
- **Inefficient testing because the project doesn't use basic control procedures such as smoke tests or configuration management.**
- **Software was not designed for testability.**
- **Excessive time spent collecting / gossiping about non-productive "metrics."**
- **Programmers send excessively buggy code into testing.**
- **Programmers don't test their own code, relying on testers instead. The more testers, the less these programmers do.**

Ratios Should Emerge from Project

- **Test groups play different roles in different companies.**
- **Testing projects vary widely in the amount of new code needed compared to the amount of testing needed.**
- **Programs vary widely in their complexity and bugginess.**
- **Markets vary in their error-tolerance.**
- **Projects differ widely in their documentation requirements.**

To justify your staff size, work from your staff's tasks.

Beware of overstaffing, it can do more harm than good.

Ratios Focus Attention on the Wrong Thing

- **Ratios focus on the relationship of labour supplied by testers to labour of non-testers.**
- **They hide essential details, like:**
  - **What tasks are being done**
  - **What efficiencies have been achieved by one group**
- **A manager focused on ratios is likely to do the management-by-objectives, let's-shave-the-ratio thing instead of focusing on the tasks at hand.**
- **"Improving" the ratio (by driving it up or down) can have a negative effect on the appropriateness of staffing.**

Factors that Affect Relative Staffing Size

- **Mission of the testing group.**
- **Allocation of labor (responsibility for different tasks) between testers and programmers.**
- **People and their skills.**
- **Partnerships between testers and other stakeholders.**
- **Product under test.**
- **Market expectations**
- **Project details (e.g. what resources are available when.**
- **Process (principles and procedures intended to govern the running of the project)**
- **Methodology (principles and procedures intended to govern the detailed implementation of the product or the development of product artifacts)**
- **Test infrastructure**

A Simple Two-Factor Approach for Estimation

		Project Size				
		Very Small	Small	Medium	Large	Very Large
Project Risk	Very low					
	Low					
	Medium					
	High					
	Very high					

In Sum

- **If you say that the ratio, on this project, was 2 to 1, there is no widespread agreement on what that means.**
- **If technical approaches change, the relative workloads change, but this isn't evident if you are thinking more in terms of ratios and less in terms of the work to be done.**
- **Ratios might sometimes be useful (for what purpose? Within what scope?)**
- **Estimators that tie more directly to the work to be done might be less subject to misunderstanding or manipulation.**