MONTHLY NEWSLETTER

PRAGMATIC S·O·F·T·W·A·R·E

Dec 2008 - Pragmatic Software Newsletters

The Seven Habits of Highly Effective Testers

Published in 1989, *The Seven Habits of Highly Effective People*, written by Stephen R. Covey has helped millions establish great habits for achieving true interdependent effectiveness in their life and their jobs. This article discusses the 7 habits, framing the habits for **highly effective testers**. Below are the 7 habits:

1. Be Proactive
2. Begin with the End in Mind
3. Put First Things First
4. Think Win/Win
5. Seek First to Understand, Then to be Understood
6. Synergize
7. Sharpen the Saw

Habit 1 - Be Proactive

A tester's goal in any software project is to ensure that the software is delivered with high quality. When software projects fail due to poor quality, you can either be proactive or reactive when analyzing what caused it. If you are reactive, you will blame other people and circumstances for problems or obstacles. If you are proactive, you will take responsibility for the failure and find ways to correct it in future projects. Upon completion of every project, your team should do a "post mortem" or "retrospective" where you openly discuss things that were done successfully in the project and things that were done poorly. Below are some ideas for being proactive on future projects:

1. **Be Responsible for Great Requirements** - Don't blame others for poor requirements. Instead, work with the team to fully analyze the requirements to ensure they are complete, accurate and testable.
2. **Analyze Traceability** - Creating a traceability matrix of test cases for each requirement allows you to analyze the test cases for coverage, testability, and completeness. Proactively hold team meetings to review your test cases to ensure you have fully understood the requirement and have adequate test coverage. Post your test cases for the development team to review before coding begins, this will reduce rework and QA time.
3. **Communicate Effectively** - During testing, it is imperative that everyone knows the status of the testing effort. Communicate daily status via email or a discussion forum. Include metrics like defect counts, requirement coverage, number of test cases run, passed, failed, and awaiting run, etc.
4. **Describe Defects Effectively** - When creating defects, spend time creating a good defect description, steps to reproduce and expected results. Include screen shots and as much information as needed to fully reproduce the issue. This will reduce QA rework.

Habit 2 - Begin with the End in Mind

Your end goal for a software project should be to deliver high quality software that meets the needs of the client. Before coding begins, you should make a list of success criteria that you judge the project on. For example, your success criteria may be that the software produces specific results, has no known defects (or a small number of low severity defects), is well documented, is easy to use, etc. By defining the success criteria up front, you can objectively evaluate whether the project met the criteria or not. Solicit help from all team members (project managers, product managers, testers, automation engineers, developers, documentation specialists, etc.) when defining the success criteria. By getting a team perspective of the success criteria, you will have better and more measurable criteria and you will get much better buy-in from the team.

Habit 3 - Put First Things First

Prioritizing your work effort is critical. You must apply effort to the most important things first, followed by less important things. For example, everyone agrees that negative testing is important to ensure that software gracefully handles scenarios where the user tries things that are not normally done and it was not designed to do. But when stacked up against positive testing, negative testing is definitely less important. So begin your testing effort by testing the software to ensure it works as designed and test it vigorously for this. Once that effort has been completed, then perform your negative testing (testing bounds, invalid data entry, overflow, injection, etc.).

Habit 4 - Think Win/Win

In many organizations, development and testing teams play a blame game and create tension between the teams. This can be very disruptive and can greatly affect the quality of the software project and the user experience. The development and testing teams should have a common goal -- to ensure that the client receives the software with the highest of quality. If this is a unilateral goal of the team, it makes sense for all team members to provide help and encouragement to each other so that when the software is shipped with high quality and the client is happy, everyone on the team basks in the joy of a happy client. If you want to encourage an environment of trust, respect and foster an win/win team, here are a few tips:

1. **Share Knowledge** - Don't hold your knowledge to yourself, share it with others.
2. **Socialize** - Eat lunch with members in different roles in your company. Learn more about them, take a general interest in their hobbies and personal goals.
3. **Encourage Others** - Offer congratulations and compliments to team members that you see are doing a great job. Tell your (and their) manager how well you think they are doing. Tell them how much you appreciate their efforts.
4. **Help Struggling Team Members** - If you see team members struggling, jump in and offer to help. If you offer, follow through and ensure they get the help they need. You may need help in the future so offering help can foster a win/win relationship for you in the future.

Habit 5 - Seek First to Understand, Then to be Understood

Many of us have a bad habit of blocking out a conversation and not listening because we so desperately want our opinion to be heard. Every tester and team member has a different experiences, different perspectives and motivations. Before you can solve any problem, it is important to first listen intently and diligently to fully understand the problem. Once you feel you have all the facts, solicit ideas for multiple solutions. Having several options can provide better discussions and allows team members to tweak initial solutions into solutions that are more far reaching and solve the problem in a more direct way. If you disagree with an approach, don't attack the person that offered the approach. Instead, explain based on your past experiences why you think there might be a better approach.

Habit 6 - Synergize

Team collaboration is the key to a synergized team. A synergized team is made up of divergent team members that have different strengths, different backgrounds and different perspectives. Encourage these differences but provide your team with tools that allow you maximize their effectiveness. Highly collaborative teams communicate with each other by sharing their calendars, posting their statuses into discussion forums so that everyone is aware of what the other is doing and accomplishing. These teams keep track of all tasks they work on each day, the number of hours worked, the number of hours remaining and variances to plan. They also share documents that illustrate best practices and produce white papers that teach others what they have learned.

Habit 7 - Sharpen the Saw

Productive testers see the need to continue honing their skills and love learning new techniques, best practices and approaches. They have a thirst for knowledge, reading every testing book they can get their hands on. They learn how to make their jobs easier -- by automating test cases and applying best practices that reduce QA time and increase software quality. They stay in touch with the testing community by visiting testing sites like Sticky Minds, QA Guild and others. They also know when to have fun. They recharge their batteries by taking great vacations and by having outside hobbies and activities.

Helpful Resources

Below are some helpful resources and templates to aid you in developing software solutions:

- **Software Planner** - http://www.SoftwarePlanner.com
- **Automated Testing Resources** - http://www.Star-QA.com
- **Pragmatic Agile Development** -http://www.pragmaticsw.com/PADOverview.pdf
- **Agile Training** - http://www.PragmaticSW.com/Services.asp
- **Software Development /QA Templates** - http://www.pragmaticsw.com/Templates.asp

About the Author

Steve Miller is the President of Pragmatic Software (http://www.PragmaticSW.com). With over 24 years of experience, Steve has extensive knowledge in project management, software architecture and test design. Steve publishes a monthly newsletter for companies that design and develop software. You can read other newsletters at http://www.PragmaticSW.com/Newsletters.asp.