# Tips for Minimizing Software Defects via Inspections

December 2003 - Pragmatic Software Newsletters

Many of us have experienced projects that drag on much longer than expected and cost more than planned. Most times, this is caused either from inadequate planning (requirement collection and design) or from an inordinate number of defects found during the testing cycle.

A major ingredient to reducing development life cycle time is to eliminate defects before they happen. By reducing the number of defects that are found during your quality assurance testing cycle, your team can greatly reduce the time it takes to implement your software project.

The key to reducing software defects is to hold regular inspections that find problems before they occur. Below is a list of **5 Tips for Reducing Software Defects**:

1. **Conduct Requirement Walkthroughs** - The best time to stop defects is before coding begins. As the project manager or requirements manager begins collecting the requirements for the software, they should hold meetings with two or more developers to ensure that the requirements are not missing information or are not flawed from a technical perspective. These meetings can bring to surface easier ways to accomplish the requirement and can save countless hours in development if done properly. As a rule of thumb, the requirements should be fully reviewed by the developers before the requirements are signed off.

2. **Conduct Peer Code Reviews** - Once coding begins, each programmer should be encouraged to conduct weekly code reviews with their peers. The meeting is relatively informal, where the programmer distributes source code listings to a couple of his/her peers. The peers should inspect the code for logic errors, reusability and conformance to requirements. This process should take no more than an hour and if done properly, will prevent many defects that could arise later in testing.

3. **Conduct Formal Code Reviews** - Every few weeks (or before a minor release), the chief architect or technical team leader should do a formal inspection of their team's code. This review is a little more formal, where the leader reviews the source code listings for logic errors, reusability, adherence to requirements, integration with other areas of the system, and documentation. Using a checklist will ensure that all areas of the code are inspected. This process should take no more than a couple of hours for each programmer and should provide specific feedback and ideas for making the code work per the design.

4. **Document the Results** - As inspections are held, someone (referred to as a scribe) should attend the meetings and make detailed notes about each item that is found. Once the meeting is over, the scribe will send the notes to each team member, ensuring that all items are addressed. The scribe can be one of the other programmers, an administrative assistant, or anyone on the team. The defects found should be logged using your defect tracking system and should note what phase of the life cycle the defect was found.

5. **Collect Metrics** - Collect statistics that show how many defects (along with severity and priority) are found in the different stages of the life cycle. The statistics will normally show over time that when more defects are resolved earlier in the life cycle, the length of the project decreases and the quality increases.

Below are some helpful templates to aid you in developing software solutions on-time and on-budget:

- **Project Management Guidelines -** http://www.PragmaticSW.com/Pragmatic/Templates/ProjectMgtGuidelines.rtf
- **Functional Specifications -** http://www.PragmaticSW.com/Pragmatic/Templates/FunctionalSpec.rtf
- **Architectural Overview -** http://www.PragmaticSW.com/Pragmatic/Templates/ArchitectureOverview.rtf

- **Detailed Design** - http://www.PragmaticSW.com/Pragmatic/Templates/DetailedDesign.rtf
- **Strategic Planning Document** - http://www.PragmaticSW.com/Pragmatic/Templates/StrategicPlanning.rtf
- **Test Design** - http://www.PragmaticSW.com/Pragmatic/Templates/TestDesign.rtf
- **Risk Assessment** - http://www.PragmaticSW.com/Pragmatic/Templates/Risk%20Assessment.rtf
- **Weekly Status** - http://www.PragmaticSW.com/Pragmatic/Templates/WeeklyStatusRpt.rtf
- **User Acceptance Test Release Report** - http://www.PragmaticSW.com/Pragmatic/Templates/UATRelease.rtf
- **Post Mortem Report** - http://www.PragmaticSW.com/Pragmatic/Templates/PostMortem.rtf
- **All Templates** - http://www.PragmaticSW.com/Templates.htm
- **Prior Newsletters** - http://www.PragmaticSW.com/Newsletters.htm
- **Software Planner** - http://www.SoftwarePlanner.com

## About the Author

Steve Miller is the President of **Pragmatic Software** (http://www.pragmaticsw.com) , the makers of **Software Planner** (http://www.softwareplanner.com) and **Defect Tracker** (http://www.defecttracker.com). With over 20 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.htm. Steve's email is steve.miller@pragmaticsw.com.

---

Pragmatic Software Co., Inc.
9085 E. Mineral Circle
Suite 340
Englewood, CO 80112

Phone: 303.471.8355
Fax: 303.346.1749
Web site: http://www.PragmaticSW.com
E-mail: info@pragmaticsw.com