An Analysis of Defect Densities Found During Software Inspections

John C. Kelly
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

Joseph S. Sherif
California State University, Fullerton, California

Jonathan Hops
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

Software inspection is a technical evaluation process for finding and removing defects in requirements, design, code, and tests. The Jet Propulsion Laboratory (JPL), California Institute of Technology, tailored Fagan's original process of software inspections to conform to its software development environment in 1987. Detailed data collected from 203 inspections during the first three years of experience at JPL included averages of staff time expended, pages covered, major and minor defects found, and inspection team size. The data were tested for homogeneity. Randomized samples belonging to the various phases or treatments were analyzed using the completely randomized block design analysis of variance ($\alpha = 0.05$). The results showed a significantly higher density of defects during requirements inspections. The number of defect densities decreased exponentially as the work products approached the coding phase because defects were fixed when detected and did not migrate to subsequent phases. This resulted in a relatively flat profile for cost to fix. Increasing the pace of the inspection meeting decreased the density of defects found. This relationship held for major and minor defect densities, although it was more pronounced for minor defects.

INTRODUCTION

This article describes an analysis of factors influencing the defect density of products undergoing software inspection. The products inspected belong to independent software intensive projects at the Jet Propulsion Laboratory (JPL), California Institute of Technology, that require a high level of quality. JPL is funded by NASA to conduct its unmanned interplanetary space program. Software inspections were introduced at JPL in 1987 to improve software quality by detecting errors as early in the developmental life cycle as possible.

Software inspections are detailed technical reviews performed on intermediate engineering products. They are carried out by a small group of peers from organizations with a vested interest in the work product. The basic process is highly structural and consists of six consecutive steps: planning, overview, preparation, inspection meeting, rework, and follow-up. The inspection process is controlled and monitored by use of metrics and checklists. Fagan gives one of the best fundamental descriptions of this process [1].

JPL tailored Fagan's original process to improve the quality of the following technical products of a software-intensive system: software requirements, architectural design, detail design, source code, test plans, and test procedures. For each of these types of products a checklist was tailored for JPL's application domain, standards, and software development environment. Supplemental tailoring included the addition of a "third hour" step to Fagan's process. The third hour step was first discussed by Gilb [2]. JPL's third hour step includes time for team members to discuss problem solutions and clear up open issues raised in the inspection meeting. Other tailoring included substantial use of

Address correspondence to John C. Kelly, Jet Propulsion Laboratory, California Institute ot Technology, Pasadena, CA 91109.

0164-1212/92/$5.00

software product assurance personnel as inspection moderators, a JPL-specific training program, and new data collection forms.

The analysis of defect densities from inspections was performed to ensure that JPL inspections meet the conditions of quality inspection, verify previous research findings on inspections, and understand the factors influencing inspection results [5]. Although it was expected that the results would agree with previous findings on inspections, some differences were observed.

METHODS

Data was collected on 203 inspections performed on five software-intensive projects at JPL. Practically all inspection team members were trained in a one-and-a-half-day course on formal inspections [3]. Software product assurance supplied 70% of the moderators. The inspections took place between February 1987 and April 1990. Although the projects used ADA, C, and SIMULA, only 16% were performed on code. Table 1 shows the types of inspections performed in this study and the sample size for each type.

The data included in this study were recorded on "Formal Inspection Detailed Report" and "Inspection Summary Report" forms (Appendixes 1 and 2). Each inspection produced a complete set of forms shown in the process diagram (Figure 1). This information was placed into a data base and monitored. Occasionally, the chief moderator would contact the inspection moderator when reports were abnormal. This was done to provide feedback for inspections which were experiencing difficulties. Eleven inspection reports were rejected for analysis in this sample because they violated some of the fundamental rules of inspections shown in Appendix 3.

Checklists were used to help inspection team members focus on what to look for in a technical work product and provide categories for classifying defects. A generic checklist was provided in the training materials for each type of inspection (R1, I0, I1, I2, IT1 and IT2). Team members may use the generic checklist or tailor this list to match their own environment and development standards. However, we encouraged inspectors to maintain the 15 main categories for types of defects shown in the Formal Inspection Detailed Report.

The metrics used to monitor and analyze inspections can be classified into three areas: staff time, types of defects, and work product characteristics. The staff time expended was recorded as total hours during each stage of the inspection process. Partway through the study we included the inspection meeting in staff time. The organizational areas represented by the author, moderator, and inspector were also recorded.

Each defect was classified by severity, checklist category, and type. Severity was considered major, minor, or trivial. Trivial defects in grammar and spelling were noted and corrected, but were not included in this data analysis nor on the Formal Inspection Detailed Report. The type of defect (missing, wrong, or extra) was recorded on the forms but not in the data base. Although this information is not as institutionally critical, it is a useful guide during the rework stage of the process.

The work product characteristics included size (by pages or lines of code), phase and type of product (requirements, test plan, etc.), and project. Since inspections were usually introduced relatively early in the developmental life cycle when most products were technical documents, the preferred size reporting metric was pages. A typical page of JPL documentation is 38 single-spaced, 10-point lines per page. A page containing a diagram was considered a page of text. Number of pages was a more accurate measure of material undergoing inspection than "estimated lines of code" for technical documents, since projects did not have a history of a detailed accounting of the second metrics during the early life cycle phases. Because most of the data were recorded in pages, we found different relationships than in previous studies. Note that "pages" is

Table 1. Types of Inspections and Defect Density per Inspection

Inspection Type		Sample Size	%	Defect Densities		
				Major	Minor	Total
RI	Software Requirements Inspection	23	11.3	6.5	23.4	29.9
I0	Architectural Design Inspection	15	7.4	2.5	16.4	18.9
I1	Detailed Design Inspection	92	45.0	3.5	10.6	14.1
I2	Source Code Inspection	34	17.0	1.1	11.5	12.6
IT1	Test Plan Inspection	16	8.0	10.3	11.8	22.1
IT2	Test Procedures & Functions Inspection	23	11.3	6.4	13.0	19.4
Totals		203	100	30.3	86.7	117.0

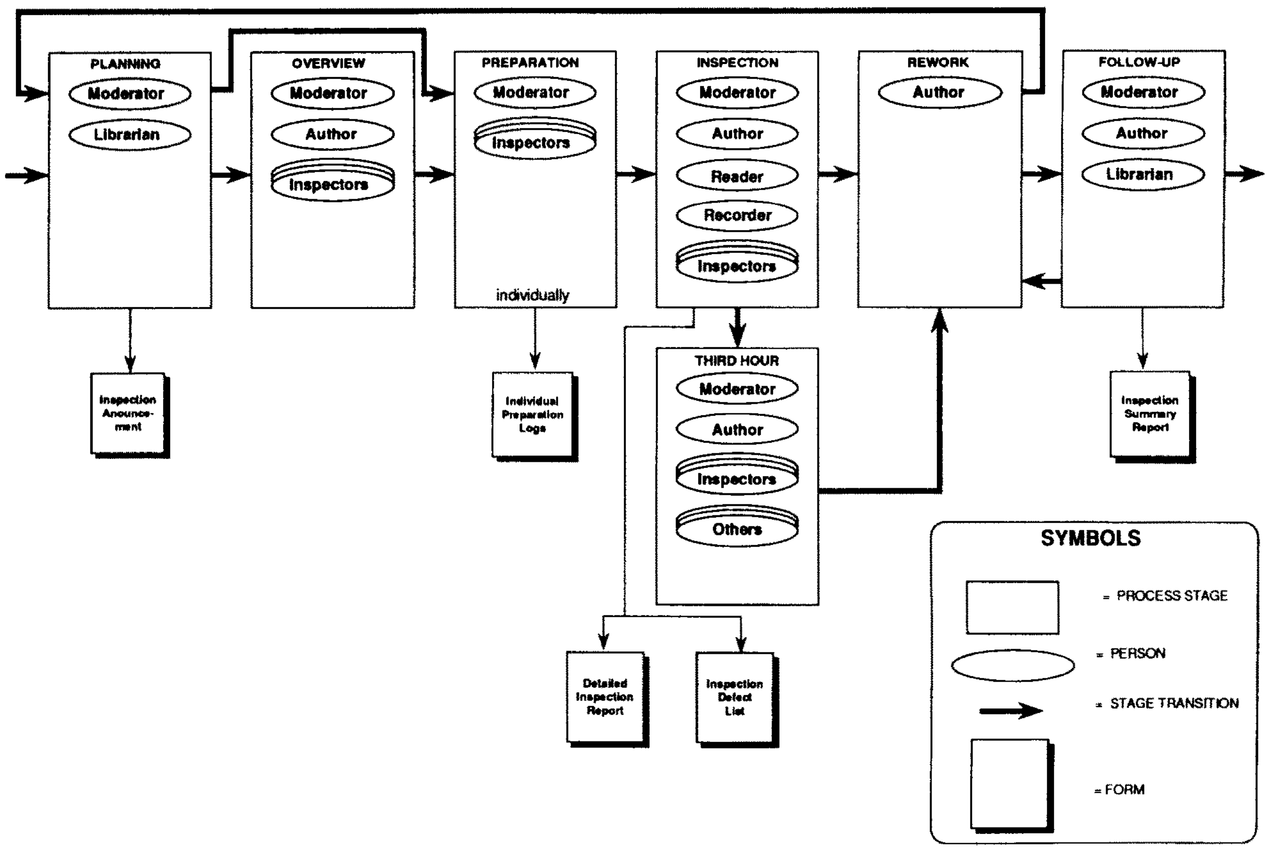
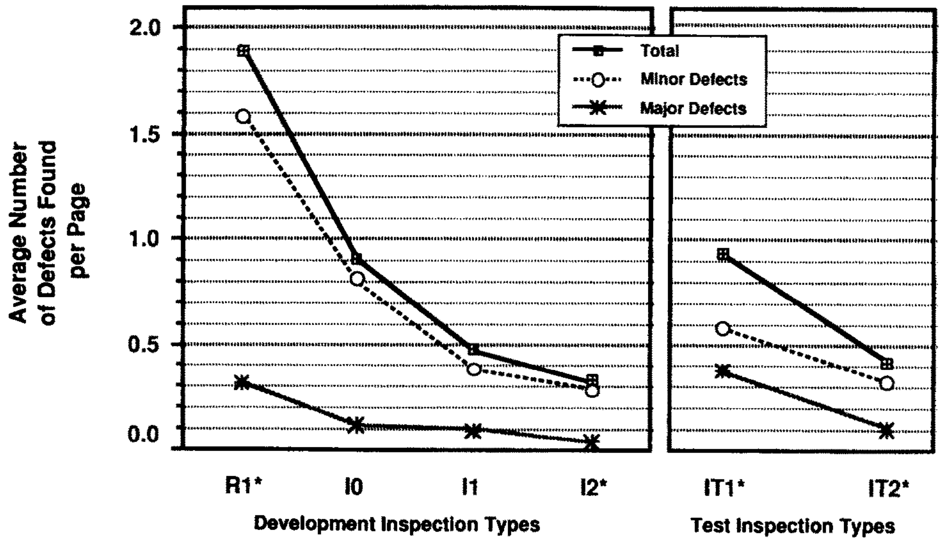


Figure 1. Overview of the software inspection process.



** At the alpha= 0.05 level of significance ANOVA F test showed a significant difference between the defect densities of R1 and I2, and between IT1 and IT2.*

Figure 2. Defect density vs. inspection types.

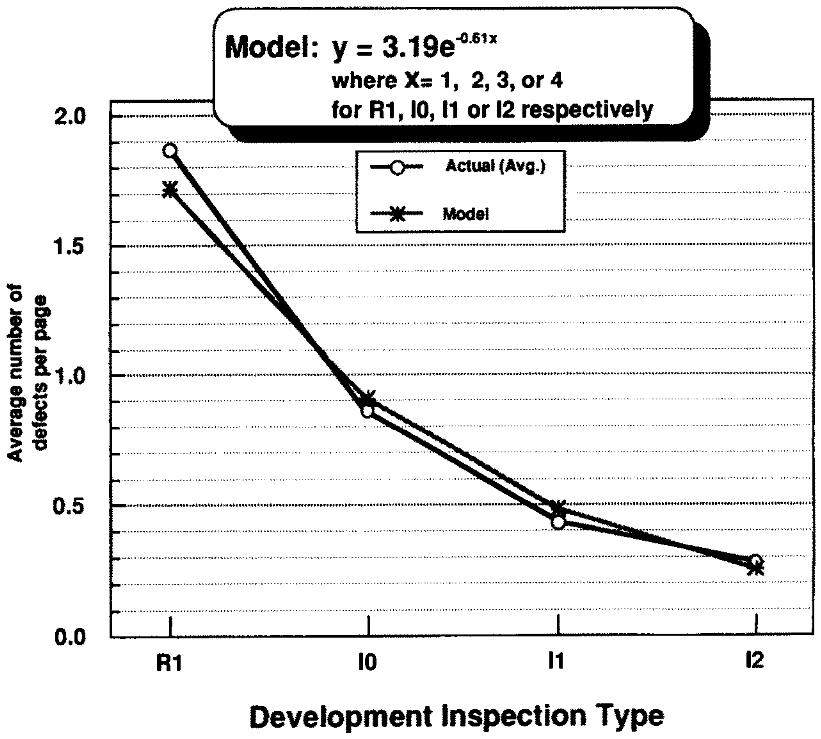


Figure 3. A developed predictive model for defect density as a function of inspection type.

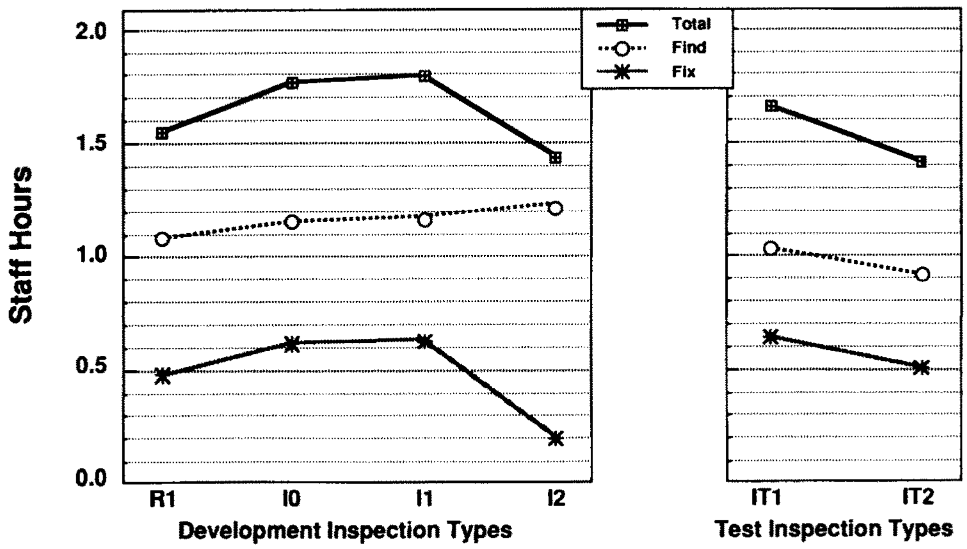


Figure 4. Staff hours per defect. Resource hours for *finding* include all time expended during planning, overview, preparation, and meeting phases. Resource hours for *fixing* include all time expended during rework, third hour, and follow-up phases. Defects include all major and minor defects.

more of a producer-oriented statistic than a product-oriented measure. One of the key metrics used in this analysis is "density of defects per page." This metric was used to compare inspections of different types and their related factors.

RESULTS

There was a higher density of defects in earlier life cycle products than in later ones. An analysis of variance was performed on data collected from the different types of inspections in the sample (R1, I0, I1, I2, IT1, and IT2). Figure 2 shows the average number of defects found per page for each of the inspection types. The defect density at the software requirements inspection (R1) was significantly higher than that of source code inspection (I2), and the defect density at test plan inspection (IT1) was significantly higher than that at test procedures and function inspection (IT2) ($\alpha = 0.05$). The defect densities found during inspections decreased exponentially as the development work products approached the coding phase Figure 3.

The number of staff hours needed to fix defects at the phase where they originated was not statistically significant ($\alpha = 0.05$); however, the cost to fix coding errors generally tended to be lower, as shown in Figure 4. Latent defects found in high-level documents were recorded as "open issues" and submitted to the change control board. Since we did not know the timely outcome from the control board, these potential defects were not tracked in this study. However, the average cost to fix defects close to their origin found during the inspection process was 0.5 hours, which is considerably less than the range of 5-17 hours required to fix defects found during formal testing, as reported by recent JPL projects [3]. This is because fixing defects during formal testing (i.e., test failure) requires detection and tracing of the defect and all associated documents, and then retesting.

Previously, inspection defect counts were found to decrease as the amount of code to be inspected increased [4]. Figure 5 shows this trend for the total sample of inspections in this study with respect to defect density per page.

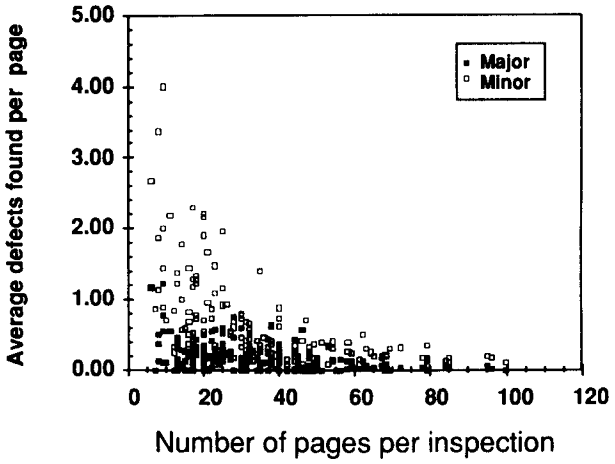


Figure 5. Inspection page rate vs. average defects found per page. Inspection meetings are limited to two hours, moderators should limit material covered to 40 pages.

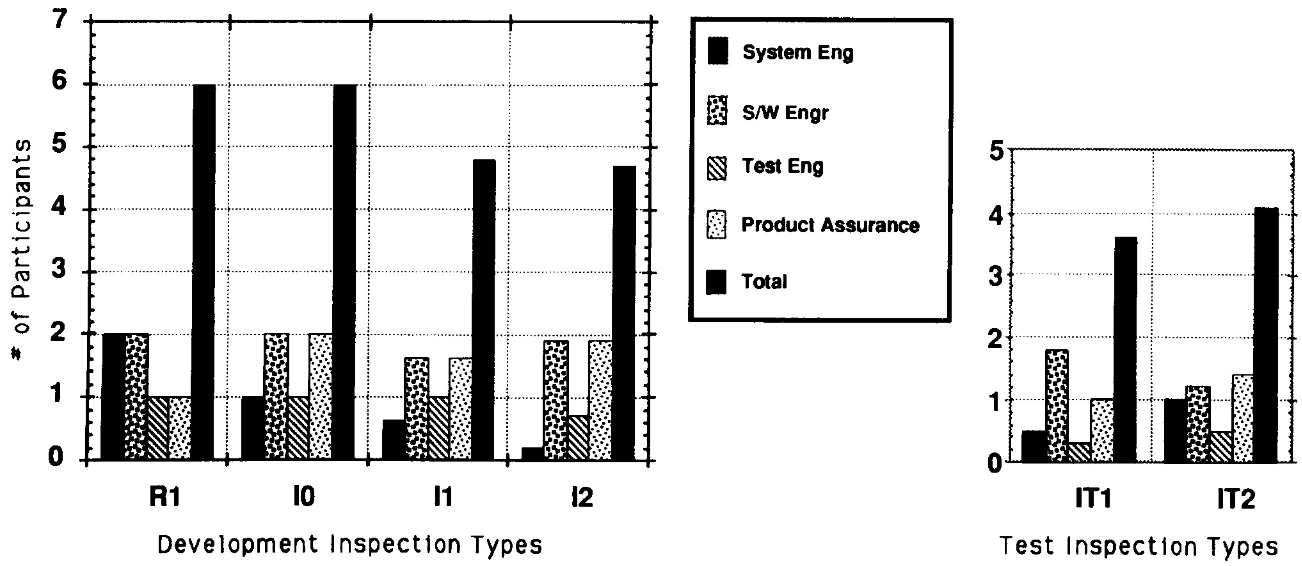


Figure 6. Team composition and size by inspection type.

Figure 6 shows the average inspection team composition and size for this sample by type of inspection. For development inspection types R1, I0, I1, and I2, the trend is to use larger teams for requirements and high-level documents and smaller teams for code. The inspection program at JPL tried to ensure that teams were comprised of members from organizations having a vested interest in the work product so as to keep inspections from being biased toward one organization's internal view of the product.

Figure 7 shows the distribution of defects percentage by types and categories. In essence, correctness, logic, and completeness constitute the highest percentage of major defects, while clarity has the highest percentage of minor defects.

CONCLUSION

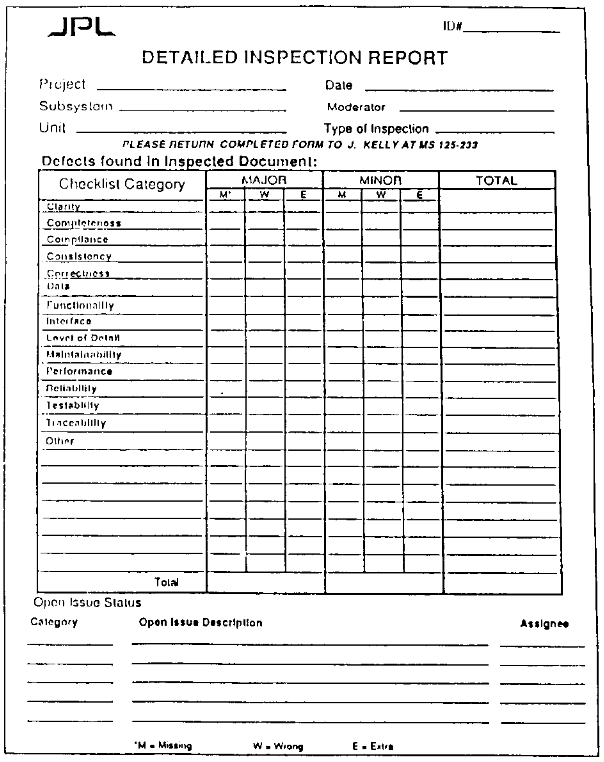
Experience has shown that formal inspection of software is a potent defect detection method; it enhances both overall software quality by reducing rework during testing, as well as maintenance efforts.

The following items highlight the results of JPL experience with formal inspections.

1. Increasing the number of pages inspected at one time decreases the number of defects found.
2. Significantly more defects were found per page at the earlier phases of the software life cycle. The highest defect density was observed during requirements inspections.
3. The cost in staff hours to find and fix defects was consistently low across all types of inspections with respect to fixing defects during testing. On average it took 1.1 hours to find a defect and 0.5 hours to fix and check it.
4. A variety of defects are found through inspections, with defects in clarity, logic, completeness, consistency, and functionality being the most prevalent.

At present, JPL recommends inspections to fulfill technical status review requirements, however, inspections are not yet mandated across the Laboratory. The results of formal inspections are very encouraging and show significant improvements in software quality.

APPENDIX 1. Formal Inspection Detailed Report.



JPL ID#________

DETAILED INSPECTION REPORT

Project ________ Date ________
Subsystem ________ Moderator ________
Unit ________ Type of Inspection ________

PLEASE RETURN COMPLETED FORM TO J. KELLY AT MS 125-233

Defects found in Inspected Document:

Checklist Category	MAJOR			MINOR			TOTAL
	M*	W	E	M	W	E	
Clarity							
Completeness							
Compliance							
Consistency							
Correctness Data							
Functionality							
Interface							
Level of Detail							
Maintainability							
Performance							
Reliability							
Testability							
Traceability							
Other							
Total							

Open Issue Status

Category	Open Issue Description	Assignee

*M = Missing W = Wrong E = Extra

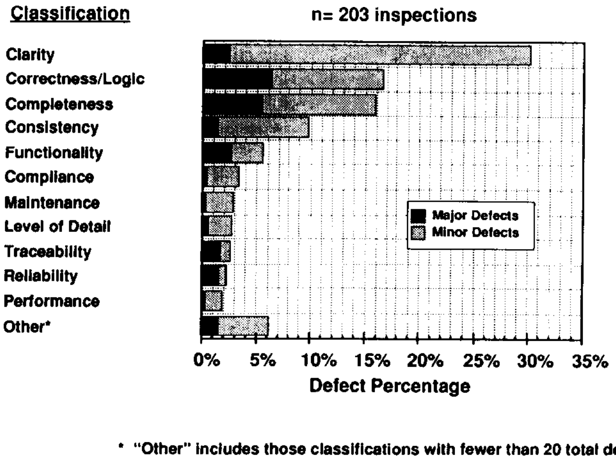
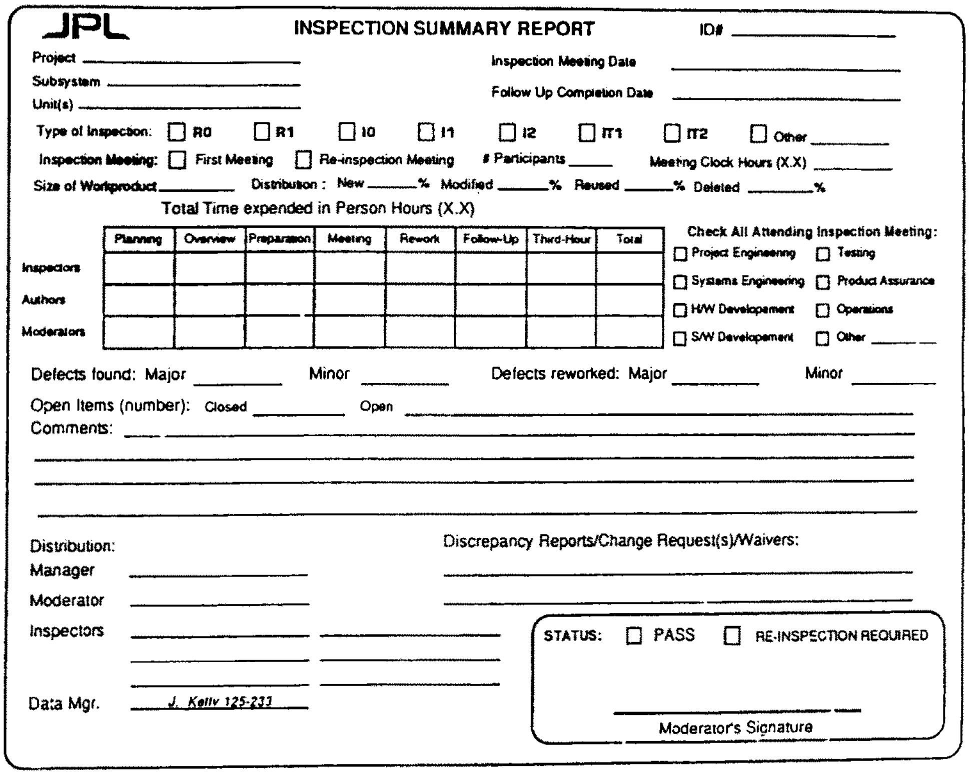


Figure 7. Distribution of defects by classification.

* "Other" includes those classifications with fewer than 20 total defects.

APPENDIX 2. Inspection Summary Report.



JPL INSPECTION SUMMARY REPORT ID# ____________

Project ____________ Inspection Meeting Date ____________

Subsystem ____________ Follow Up Completion Date ____________

Unit(s) ____________

Type of Inspection: ☐ R0 ☐ R1 ☐ I0 ☐ I1 ☐ I2 ☐ IT1 ☐ IT2 ☐ Other ______

Inspection Meeting: ☐ First Meeting ☐ Re-inspection Meeting # Participants ____ Meeting Clock Hours (X.X) ______

Size of Workproduct ______ Distribution : New ____% Modified ____% Reused ____% Deleted ____%

Total Time expended in Person Hours (X.X)

	Planning	Overview	Preparation	Meeting	Rework	Follow-Up	Third-Hour	Total
Inspectors								
Authors								
Moderators								

Check All Attending Inspection Meeting:
☐ Project Engineering ☐ Testing
☐ Systems Engineering ☐ Product Assurance
☐ H/W Developement ☐ Operations
☐ S/W Developement ☐ Other ______

Defects found: Major ______ Minor ______ Defects reworked: Major ______ Minor ______

Open Items (number): Closed ______ Open ______

Comments: ____________

Distribution: Discrepancy Reports/Change Request(s)/Waivers:

Manager ____________

Moderator ____________

Inspectors ____________ ____________

Data Mgr. J. Kelly 125-233

STATUS: ☐ PASS ☐ RE-INSPECTION REQUIRED

Moderator's Signature

APPENDIX 3. The 10 Basic Rules of Inspections.

1. Inspections are carried out at a number of points inside designated phases of the software lifecycle. Inspections are not substitutes for major milestone reviews.
2. Inspections are carried out by peers representing the areas of the life cycle affected by the material being inspected (usually 6 people). Everyone participating should have a vested interest in the work product.
3. Management is not present during inspections. Inspections are not to be used as a tool to evaluate workers.
4. Inspections are led by a trained moderator.
5. Trained inspectors are assigned specific roles.
6. Inspections are carried out in a prescribed series of steps (see Figure 1).
7. Inspection meetings are limited to two hours.
8. Checklists of questions are used to define the task and to stimulate defect finding.
9. Material is covered during the inspection meeting within an optimal page rate range that has been found to give maximum error-finding ability.
10. Statistics are kept on the number and types of defects and the time expended by engineers on the inspections.

ACKNOWLEDGMENTS

We thank the editor-in-chief of *JSS* and two anonymou referees for helpful comments and suggestions.

REFERENCES

1. M. E. Fagan, Design and Code Inspections to Reduc Errors in Program Development, *IBM Syst. J.* 1 182-211 (1976).
2. T. Gilb, *Principles of Software Engineering Manage ment*, Addison-Wesley, Reading, Massachusetts, 1987.
3. J. C. Kelly, Formal Inspections For Software Develop ment, Software Product Assurance Section, technical re port, Jet Propulsion Laboratory, California Institute o Technology, Pasadena, California, 1987.
4. F. O. Buck, Indications of Quality Inspections, IBM technical report TR 21-802, 1981.
5. Y. S. Sherif, *Systems Modeling and Computer Simulation* (N. A. Kheir, ed.), Marcel Dekker, New York, N.Y., 1988.