1990

Inspections: Software development process for building defect free software applied in a small-scale software development environment

Patricia A. Sherwood

Rochester Institute of Technology
School of Computer Science and Information Technology

Inspections: Software Development Process for Building Defect Free Software Applied in a Small-Scale Software Development Environment

By
Patricia Ann Sherwood
February 13, 1990

A thesis, submitted to
The Faculty of the School of Computer Science
and Information Technology,
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science.

Approved by: Jeffrey A. Lasky

Approved by: Name Illegible

Approved by: Name Illegible 2/19/90

April 24, 1990

Title of Thesis:

Inspections: Software Development Process for Building Defect Free Software Applied in a Small-Scale Software Development Environment

I, Patricia Ann Sherwood prefer to be contacted each time a request for reproduction is made. I can be reached at the following address: 3125 West B Ave, Plainwell, MI 49080.

I ABSTRACT

Inspections is a software management technique designed to produce higher quality software and improve programmer productivity. These improvements are achieved through rigorous examination of the products in each phase of the software development life-cycle. While the process has generally been applied to large-scale mainframe projects, this research demonstrates successful implementation of inspections in a small-scale, micro-processor based software development project.

II KEY WORDS

Inspections, Project Management, Quality Assurance, Software Development, Software Engineering, Software Quality Testing, Walkthrough.

III COMPUTING REVIEW SUBJECT CODES

Primary Code:	D.2.9	Management - Life cycle
Secondary Code:	K.6.1	Project and People Management - Life cycle
Secondary Code:	K.6.3	Software Management - Software Development
Secondary Code:	K.6.4	System Management - Quality Assurance

TABLE OF CONTENTS

1.0 INTRODUCTION AND BACKGROUND

The inspection process is a software development management technique designed to produce higher quality software and increase programmer productivity by rigorously examining the products of each lifecycle phase. The process was developed at IBM Kingston in 1972 by Michael Fagan (Fagan, 1986). Originally applied to hardware logic, the inspection process has since been used primarily in software design and coding. Available data indicates that since its introduction, inspections have primarily been used in large system (usually mainframe) projects.

This thesis is apparently the first reported work which demonstrates that the inspection process can be successfully adapted for use in a small-scale software development environment. (Also see Bisant, 1989 for results of a controlled experiment using two-person inspection teams in an academic environment.)

The development team was composed of 3 members, the project leader and two engineers. The project leader was responsible for product definition, specification development, initial design, systems test plan, production, and management of the implementation and testing processes. The engineers were responsible for detailed design, programming and unit testing.

The adapted process was applied to the development of the Computer Interfaced Commercial Items Dispenser (CICID), a dual Intel 8031 (8 bit microprocessor) based product developed by the Standard Register Company, Financial Equipment Development Department. This product is used by banking institutions to produce protected and signed Checks and Money Orders.

The CICID receives check and voucher data which would normally be sent to a standard printer. It parses the input data according to the declared format, prints the voucher and check using copyrighted protection methods and prints a digitized signature. If desired, a journal listing may be produced for each document printed. The flexibility of the software provides parsing for practically any ASCII data stream which does not contain graphics or specialized control codes.

Software for the slave 8031 processor, which supported keyboard scanning, the 40 character segmented display and a thermal journal printer had been previously developed. All software for the master 8031 processor, which supported configuration and setup, message prompting and document printing was developed using the inspection process and contained no previously written code.

The CICID is the newest product in the check dispensing equipment line. Products developed previously included a manual entry Official Items Dispenser, a Money Order Dispenser, a Gift Certificate Dispenser and a Computer Interfaced Official Items Dispenser. The software development efforts for each of these products largely occurred in an undisciplined manner only. Typically, requirement documents were incomplete, covering the physical aspects and a few key features for specific customer applications. Engineers then produced multiple iterations of the software in a "code and fix" approach until the product satisfied the specific customer.

Problems with this type of software development included:

1. Difficulty in responding to market demands. Since the initial software product was not designed to be flexible or reusable, limited portions of the code could be used from one application to the next. Software was continuously rewritten to satisfy subsequent customers.

2. No overall system coordination. As projects grew in size and in the number of people assigned, many of the low level software routines were coded multiple times. This significantly increased the size of the code and when the limited code space was exhausted forced much of the software to be rewritten.

3. Difficulty in assessing product availability and completion dates.

Software verification was also a problem. Unit testing was performed at the discretion of the individual programmer. Systems testing was performed, without test plans, by temporary, non-technical testers. They were instructed to run the machine and report problems to the engineers.

Problems with the testing process included:

1. No formal test plans or test procedures. The thoroughness of the testing relied upon testers who had limited comprehension of the machine and how it should work.

2. Logs were not produced detailing the areas covered or problems found during testing.

3. Problems discovered during testing were occasionally overlooked because the problems were not documented and their fixes verified.

4. Programmers were often interrupted to verify the machine actions as a problem.

The development and testing issues have become more pronounced as the software development group has grown. In response, a first cut software development model using inspections was suggested for use in the CICID project. Before implementation of the inspections, an intensive training program was developed and implemented to provide common development techniques and to introduce the concepts of inspections.

All engineers in the software development department attended the training sessions which covered:

- Characteristics of Good Functional Decomposition
- Yourdon Data Flow and State Transition Diagramming
- Structured Unit Testing
- Software Quality Overview
- Project and Programming Standards
- Inspections: A Software Lifecycle Methodology

After completion of the training, inspections were implemented on the CICID in each of the following lifecycle phases: Specification, Initial Design, Detailed Design, Code and Unit Test, and Systems Test.

The specification phase produced a detailed specification. At a minimum the specification included: a system description containing sufficient detail to describe how the system would work, known data requirements, system security considerations, environmental and operational constraints, and a reference to related materials. For some project, a system diagram describing the sequence of operation and device interactions or estimates of the next phase costs and schedule, would also be included.

The Initial Design described how the specification would be implemented. Data flow diagrams were the medium for description. In this phase, all major modules in the system are defined in terms of their inputs, outputs and functionality. Throughout the project, many changes were made to the system organization, due to compiler and overlay constraints. After the project was completed, the data flow diagrams were updated to show the final module organization.

Detailed Design provided the implementation details required to code a functional area in cases where the specification and initial design did not provide sufficient detail for subsequent coding. Modules from the Initial Design were further decomposed where required into a hierarchy of functions.

All source code, produced in Intel PL/M 51 high-level language, was inspected. The code inspection proved to be especially important for functional areas which did not have a detailed design inspection. Even though certain functions may have been considered to be straight forward implementation, the examination of the code which implemented these function revealed errors.

The system test plan was produced from the system specification. Two systems test plans covering different areas were developed for the project. Additionally, a user simulation test using six PC accounts payable and payroll packages was performed.

2.0 PRIOR WORK

The results of several studies of the implementation of inspections have been published. This section included descriptions of the studies and their major results.

STUDY	MAJOR RESULTS
TITLE: An IBM Comparison of the Walk-Through Method VS the Inspection Method (IBM, 1977)	An estimate of 94 programmer hours were saved per thousand lines of code as a result of the detailed design inspection, and 51 hours by the code inspection. This translates into a 23% increase in coding productivity. Equivalent testing between post unit test and systems testing showed the inspected products to contain 38% fewer errors.
TITLE: Some Experiences in the Use of Inspection Teams in Application Development (Crossman, 1979).	Using only code inspections a 95% reduction in maintenance costs was achieved.
BACKGROUND: Experiences of The Standards Bank of South Africa after using inspections for 4 years and inspecting about 1,000,000 Lines of Cobal Code.	For a series of 6 programs with approximately 143,000 lines of code, only 22 defects were found in use. They also felt that even if there was not a cost justification for continuing to use the inspection technique, that the benefits of improved service to the users and the creation of a more professional development environment justified its continued use.
TITLE: Software Inspections and the Industrial Production of Software (Ackerman, 1984).	Twenty-four hundred software developers and managers had been trained in the use of inspections. Each project is a separately managed entity. Of 42 projects, 25 are using inspections, 4 are in trial and 5 are in training.

STUDY	MAJOR RESULTS
BACKGROUND: Results of the software engineering technology transfer program to implement software inspections at Bell Laboratories.	Instead of following defect type checklists for the inspection phases, Bell stresses understanding the material during Preparation and provides preparation guidelines. They allow some defects to be listed as trouble reports which are addressed in the next development cycle. Where inspections are consistently used; developers and managers were enthusiastic and supportive of the inspection process and felt that inspections increased the product quality. The majority perceived no delay in product schedules by using inspections.
TITLE: Application of Software Inspection Methodology in Design and Code (Buck, 1984). BACKGROUND: Discussion of results of applying the inspection process to software development at IBM in Manassas, Virginia for the US Navy. Inspections performed at High Level and Detailed Design and Coding phases. Data analyzed by Software Quality Assurance group.	Newly developed code has a range of 8 - 12 defects per 1000 lines of code. Man-hours expended per major defect detected is 3 - 5 hours Data evaluated from one phase inspection to the next. If the progression of product quality did not improve, then management would investigate.
TITLE: IBM IMS Maintenance and Enhancements (Fagan, 1985) BACKGROUND: Study on use of inspections on the IBM IMS maintenance and enhancements for 1976 through 1978.	Showed a defect volume reduction of 36% and a 15% reduction to support a line of code.

STUDY	MAJOR RESULTS
TITLE: IBM VM Systems Programming Organization (Fagan, 1985).	The number of defects per thousand lines of code has been reduced by two-thirds through use of inspections.
TITLE: The Effects of Software Inspections on a Major Tele-Communications Project (Graden, 1986).	Collected three types of data: Project Management Data - size of each component, the effort expended (estimated and actual), Inspection Data - product and process data, Fault Data - errors found in testing and use.
BACKGROUND: Examination of inspection implementation at Network Software Center (NSC). All technical staff (more than 300) was trained in the techniques. Separate departments are responsible for the planning, engineering, development, testing, installation and support of the software.	Studied inspection characteristics which were critical to the end-product quality of the software. Determined fault density - faults and severity normalized by size after integration testing. Found that software subsystems with errors per inspection different from the process mean have the highest fault and severity densities. Used this fact to predict early in development which subfunctions required additional attention.
TITLE: A Code Inspection Model for Software Quality Management and Prediction (Christenson, 1986)	Model used to estimate remaining code density and thus predict which code should be reinspected.
BACKGROUND: An attempt to relate input parameters such as inspection effort to the resulting quality of code produced at AT&T Bell Laboratories.	Showed that more inspection effort is justified when the code error density is high. The cost of over inspecting is minimal in comparison to the cost of errors found in subsequent phases or in use. Policy developed - inspect code with minimal effort in order to estimate the initial error density and then reinspect with the appropriate effort. A cost savings of 40% is estimated with this policy.

STUDY	MAJOR RESULTS
TITLE: The IBM Amplification Model (IBM, 1987)	In this model 50% of the specification errors are amplified by a factor of 3.5 in the design phases and one-third of the design errors are amplified by a factor of 5 in the coding phases.
TITLE: Inspections: The Heart of Quality Control (Staton-Reinstein, 1989).	The Information Systems (IS) departments in the study ranged in size from 1000 to 3000.
BACKGROUND: A case study of four successful implementations of inspections included two large financial services companies, a major bank, and a telecommunications company	In most cases inspections use was driven by a quality assurance group outside of the software development organization. Each group demonstrated that the process found errors before testing and improved the final product. Even though a variety of data categorization, collection and reporting methods were used, each company was able to demonstrate the benefits of inspections.
TITLE: Software Inspections: An Effective Verification Process (Ackerman, 1989).	Few organizations would share their findings with the group unless they had been previously published. Survey examined inspection use in 15 organizations
BACKGROUND: Compilation of a preliminary survey on inspection use in the industry.	The data showed that inspections do not eliminate testing. They are two to ten time more efficient in removing defects than testing. Also, regardless of application or language, inspections find from seven to 20 major defects per thousand lines of code at a cost of one to five man-hours.

STUDY	MAJOR RESULTS
TITLE: Formal Inspection Review Techniques: Experience & Results (Kelly, 1989). BACKGROUND: Experience and results of implementing inspections at the Jet Propulsion Labs, CIT. Information obtained from presentation notes from the Technical Proceedings at Quality Week 1989 in San Francisco California.	Total number of inspections since March 1988 - 170. Number of projects using inspections is eight. Average Major Defects per inspection was 4.8. Average Minor Defects per inspection was 12.8. Average Number of Participants per inspection was 4.8. Work Hour per defect found: Detailed Design - 1.9 Code - 1.3 Test Plans - 1.6 Detail Test Plans - 1.7 Average - 1.7 Found that inspections were good at finding clarity, completeness, correctness, and consistency categories of defects. Inspections were cost effective for all types investigated. Higher level inspections were significantly more cost effective than lower level inspections.
TITLE: A Two-Person Inspection Method to Improve Programming Productivity (Bisant, 1989). BACKGROUND: Examined the effectiveness of 2 person inspection teams on novice productivity in an undergraduate Data Structures course. Thirteen completed the inspection assignments, nineteen completed the control (non-inspection) assignments.	For each inspection, 10 minutes was spent in how to perform inspections and what types of errors to look for. Twenty minutes was spent inspecting each product. The results indicate significant improvement in the group performing inspections. The slower the programmer, the more improvement observed.

Broadly speaking, these studies show trends in the use and results of inspections. Most applications of inspections occur in large organizations which develop massive systems. Some of the implementations are managed and measured by quality assurance groups outside of the software development group. Organizations which have established databases and procedures study the critical inspection characteristics to determine predictive measures to show which modules need further attention. IBM has developed a model which shows how defects not identified in earlier phases are amplified in subsequent phases.

Some studies show an increase in programmer productivity, others a reduction in cost and/or defects per thousand lines of code. Inspections were shown to be significantly more effective in removing defects than testing. It appears that regardless of application or language, inspections find from 7 to 20 major defects per thousand lines of code at a cost of 1 to 5 man hours.

3.0 IBM INSPECTION METHOD

Inspections can be characterized as follows:

1. Separate scheduled activities in the project development plan. The schedule allows time for reworking of problems found during the inspections.

2. Each phase in the development lifecycle produces a product which is inspected. Each phase has specific exit criteria defining when the current phase is completed and when the next phase may begin.

3. A specially trained moderator schedules, conducts and chooses the participants in the inspection.

4. The inspection technique emphasizes the accumulation and analysis of data about the types of errors and their frequency. As a data base of information concerning the inspections is collected, error trends can be analyzed. Also, the checklists used when preparing for the inspection can be updated to insure all reasonable questions have been considered during the inspection.

INSPECTION PROCESS

PROCESS OPERATIONS	OUTPUT (+ DETAILED EXIT CRITERIA)	LEVEL OF FUNCTION	ORIGIN OF TEST LEVEL
Level 0	Statement of Objectives	Component	
Level 1	Architecture	Component	
Level 2	External Specifications	Function	Test Plan
Level 3	Internal Specifications I_0 Inspection	Module	IT_1
Level 4	Logic Specifications I_1 Design Complete Inspection	Logic	Test Cases
Level 5	Coding/Implementation I_2 Code Inspection Unit Test	Logic	IT_2
Level 6	Integration Test	Function + Component	
Level 7	System Test	Component	

Figure 1. The Inspection Process (Fagan, 1976)

Process operations are the phases or levels into which the development lifecycle have been broken. Each level produces a product which is listed in the output column.

I_0, I_1, I_2, IT_1, IT_2 each indicate a point at which an inspection is performed. Before the inspection may take place, specific entry criteria must be satisfied.

Progression from one level to the next is accomplished by the product of the previous level meeting the exit criteria.

Origin of test level shows that the system test plan is written from the specification and test cases are determined from logic.

Detailed Description of Inspections:

Software inspections provide a rigorous examination of programming materials. e.g., Specifications, Design Documentation, Code Listings and Test Plans. The ideal rate of review for an inspection varies with type and difficulty.

The inspection activity is embedded in a six stage process.

- PLANNING
- OVERVIEW
- PREPARATION
- INSPECTION
- REWORK
- FOLLOW-UP (IBM, 1987)

PLANNING:

- The moderator verifies that the materials meet the entry criteria.
- The moderator selects the participants.
- The inspection is scheduled.
- The developer distributes copies of the inspection materials.

OVERVIEW:

- Developer provides an overview of the materials to those who are to participate in the inspection.
- This step may be waived by the moderator if the materials are well known to the participants.

PREPARATION:

- The participants prepare for the inspection by studying the materials. The objective is for all participants to become thoroughly familiar with the inspection materials in order to maximize inspection efficiency and effectiveness.
- The participants use checklists which suggest areas to examine for errors. The checklists have general and project specific elements. For example, while examining the materials, they:
  - Check that the materials for the product being inspected match materials from the previous phase(s).
  - Understand the required inputs and expected outputs (external linkages to and from each module).
  - Understand the data environment of each module.
  - Comprehend the control flow and logic (where applicable).
  - Check that the exit criteria have been met.
  - Note discrepancies or errors found so that they may be recorded during the inspection meeting.

- No attempt is made during this step to find solutions to any problems uncovered. That is the function of the rework step.

INSPECTION

The inspection team attends the inspection meeting. Since the meeting is a working session with a fixed objective (to find errors) others are not encouraged to attend.

- The purpose is to find errors, not solutions.
- The reader paraphrases the material being inspected.
- The group examines the product.
- All errors discovered in the inspection are recorded and classified according to:

    . severity: major, minor,
    . category: missing, wrong and extra,
    . type: general and project specific.

At the end of the meeting, the list of errors is reviewed to make certain that the list is complete.

REWORK

Within one day of the inspection, the moderator distributes the inspection detail report which summarizes the problem list.

- All errors found in the inspection must be fixed.
- The developer reports back to the moderator when all problems have been fixed.

FOLLOW-UP

After all errors and ambiguities have been corrected, and if a reinspection has not been scheduled, the moderator verifies the completeness and accuracy of the reworked materials and gives formal approval. This allow the development effort to move forward.

- Moderator determines all problems have been fixed and the exit criteria have been met.
- All rework must be completed and verified before moving on to the next phase of the software lifecycle.
- If more than 5% of the materials were changed, they must be reinspected.
- Changes to previous levels are noted (as needed).
- Results of the inspection are included in the data base.

PARTICIPATING IN AN INSPECTION:

An inspection is held when the moderator concurs with the author that the entry criteria can be met. An overview of the materials is then scheduled for the participants (if necessary). After the overview has been presented, the inspection materials are handed out and a time for the inspection is agreed upon.

To aid the participants each type of inspection has a checklist of items to look for when preparing for the inspection. The lists are application specific and are another area where the inspection process is adapted to fit the application.

The number of participants must be limited. They should assume the following roles:

- Moderator
- Author
- Tester
- Reader

Additional participants may be added if there are complex interfaces or if other persons are affected by the product being inspected.

The moderator should be a senior person on the project. The moderator's responsibilities include:

- Verifying that the inspection material meets entry criteria
- Helping in preparing the list of participants
- Determining that each person has adequately prepared for the inspection
- Recording the list of defects
- Maintaining the inspection focus on describing the defect not fixing it
- Verifying the fix with the author when the number of rework items is small (less than 5%)
- Deciding if another inspection is required after the defects have been fixed
- Signing off the materials inspected, signifying that the exit criteria have been met

The author has the following responsibilities:

- Provide material which will meet entry criteria
- Prepare an overview (if required)
- Distribute copies of the materials to all participants
- Fix all defects

The tester has the responsibility of writing the system test plan. The primary responsibility of this individual is to determine the testability of the product. He/she will also keep track of the inter-module control and data links during the inspection.

The reader reads aloud each line of the design/code and substitutes values for variables as the material is reviewed. All participants review the product for errors.

If possible, the inspection team should be limited to these 4 roles. The roles should be filled by those who interface with the module being inspected.

The data collected during the inspection should be analyzed so that the process can be improved. The analysis may indicate an update to the entry/exit criteria or checklist items. Testing results should also be analyzed to determine where defects should have been found and how the process can be improved to find the defects before testing.

It is imperative that changes at every phase be controlled. Any change after an inspection sign-off degrades the quality of the final product. Changes must be desk checked and run through unit testing if less than 6 Non-Commentary Source Statements (NCSS) are changed. If 6-40 NCSS are affected, the process is started from the code inspection phase. When more than 40 NCSS are affected the process is restarted at the design phase. Anything greater than 33% of the NCSS must start at the specification phase. (Fagan, 1985)

These rules apply to maintenance as well as fixes, although, the complete process may be utilized where appropriate. New features should always follow the entire process.

4.0 THE INSPECTION PROCESS AT STANDARD REGISTER

The IBM inspection model formed the basis for Standard Register's inspection process. The relative size of the CICID project required changes in the size of the inspection team, roles played by the team members, reinspection requirements, entry and exit criteria for each level, lifecycle phases, and types of inspections performed.

The development team participated in each inspection. The specification inspection was also attended by the department manager, marketing representative (our customer), and programming engineer of a similar product. The lead tester was included in the inspection of the system test plan.

Inspection roles were limited to Moderator, Developer, and Inspector(s). The IBM inspector roles of Moderator, Developer, Tester, and Reader were not followed due to the limited size of the development team. The developer generally read through each line of the product or described the data flow (if a diagram was being inspected). The moderator recorded each error or problem identified and maintained the focus and direction of the inspection. Other participant(s) in the inspection assisted in pointing out errors, inconsistencies and other problems related to their area of responsibility.

Within 2 days of the inspection, the categorized problem list was distributed by the moderator. The moderator made certain all problems had been addressed before moving to the next phase. At various points in the project the specification was updated to reflect the changes made.

The IBM method requires reinspection when 5% of the lifecycle phase product changes. In comparison, products of lifecycle phases were reinspected at Standard Register when the solution required major changes to logic or implementation. There was no percentage threshold for determining reinspection. Also, certain code segments were reinspected just prior to systems testing, as some problems identified in unit testing required changes in each engineer's code.

Changes made to the entry and exit criteria were primarily application dependent. (See Appendix A for detailed phase description and entry/exit criteria.)

Inspections were initially implemented for the 5 stage lifecycle described earlier. Movement from one level to the next was accomplished by the product fulfilling the exit criteria for that level. Description of products subject to inspections is included on page 4.

It was evident early in the implementation that three changes would need to occur so that the use of inspections at Standard Register would be viable. The first change was prompted by the realization that not all program functions needed to go through every phase in order to produce a high quality product. After the Specification and Initial Design many functions could be directly coded, skipping the Detailed Design phase. The decision of which functions would bypass the detailed design was made by the project leader in conjunction with the engineers. In general, the areas which did not require Detailed Design were user interface and executive options which determined configuration parameters.

The second change affected the production of formal unit test plans. Examination of the formal unit test showed that it took more time to produce the unit test and inspect it than it did to develop the design and code for the function. It also did not find problems in the code that an informal unit test would have found. Therefore, after one unit test inspection the formal unit test plan was dropped from the lifecycle. The developers were however, required to unit test their code informally before sending the product to systems testing.

This change is consistent with the Ackerman study at Bell Laboratories. "In practice, it is extremely difficult to specify input sets that give 100 percent branch coverage: but detailed design and code inspections are a line-by-line examination of designs and code and hence provide 100 percent coverage since all branches are considered...rational judgements can be made on which to use (inspections or unit testing), or how they should be combined." (Ackerman, 1984).

The final change, adding a Requirements phase for new projects, was determined after the project completion. Since the customer (Marketing) does not provide formalized requirements, the engineering organization needs to obtain agreement on a general direction for a new project before proceeding with the specification.

The requirement phase would include a one to five page document describing the product in broad terms. The Requirements document should be inspected but the errors should not count in the final totals. During the requirements phase it would be difficult to consider points discussed as errors since this is the time to formalizing ideas of how the product is going to look.

The lifecycle phases in use at Standard Register after completion of the CICID project are:

Level 0:	Requirements	Required for New Products
Level 1:	Specification	Required
Level 2:	Initial Design	Required
Level 3:	Detailed Design	Optional - Used when insufficient detail exists to code functional area.
Level 4:	Code	Required
Level 5:	Systems Test	Required

This lifecycle differs from the IBM method in the addition of Requirements and Specification phases and the combination of two system test plan phases, IT1: Test Plan and IT2: Test Cases into one Systems Test phase.

5.0 DATA COLLECTION AND ANALYSIS

Eighteen inspections were held during the project. Of these, 6 were reinspections. Time spent in preparation and during the inspection meeting was recorded for each inspection. Errors discovered in the inspections were assigned a problem type, a severity and a designation of missing, wrong or extra. Errors discovered during testing and changes made to the specification were categorized in the same manner.

5.1 CATEGORIZATION OF INSPECTION ERRORS

A specific list of problem types was developed for each lifecycle phase. The errors were assigned to a problem type and further broken down into severity and category.

Severity was determined using the following guidelines:

Major:

- Errors which would be difficult to find or fix at a later time.
- Errors which would cause serious problems in operation.
- Sections left undefined (or not implemented) which are critical to operation.
- Significant deviations from previous phase documents.

Minor:

- Comments which are missing or not correct.
- Typographical errors in coding
- Lessor non-mandatory deviations from previous phase documents.
- Lessor implementation details which are not correct.
- Errors which would not be difficult to find or fix later.
- Errors which cause minor problems in operation.
- Most operator interface prompting problems.

Categories of Missing, Wrong and Extra were also assigned to each error. Assigning categories seemed to have limited significance and added little to the analysis of errors on this project.

Assigning an error to a problem type was fairly straight forward. Most errors fit a problem type defined at the beginning of the process. Of the 5 phases and 359 errors only 16, or 4.4% of the errors were assigned to the Other problem type.

5.2 PROJECT DURATION

The project was started on September 19, 1988 with the initial inspection of the specification and ended on March 10, 1989 with release to the Beta Site. Total elapsed time was 24 weeks.

ERRORS FOUND AND INSPECTION TIME (IN HOURS)

PHASE	PREP TIME	INSPECT TIME	TOTAL TIME	ERRORS MAJ	ERRORS MIN	ERRORS TOTAL	TIME PER ERROR
SPEC	43.5	54.5	98.0	8	56	64	1.5
INITIAL DES	2.0	6.0	8.0	1	19	20	0.3
DETAIL DES	5.5	16.5	22.0	16	34	50	0.4
CODE	33.5	54.0	87.5	23	177	200	0.4
SYSTEM TEST	4.5	12.0	16.5	2	23	25	0.7
TOTAL	89.0	133.0	222.0	50	309	359	0.6

Major Findings:

A total of 2,211 development hours were charged to the project. Inspection preparation of 89.0 hours comprised 4% of the project. The inspection meetings consumed 133.0 hours for 6% of the project. In total, 10% of the project time was spent in the inspection process.

The percentage of inspection and preparation time in the CICID project is at the lower to middle point of the range of percentages reported by others. The summarization of the available data on inspections by the Institute for Zero-Defect Software indicated that inspections take 4% to 15% of the project time (Ackerman, 1989). The Standard Bank of South Africa Limited, estimates a 10% overhead for inspection work on their projects (Crossman, 1979). Fagan has indicated that typically 15 - 20% of an inspected project's time is spent in inspections (Fagan, 1985).

The time per major error of 4.44 hour (222 hours / 50 major errors) is at the high end of the 3 - 5 hours per major error reported at IBM Manassas (Buck, 1984) and 1 - 5 hours per major error described by Ackerman (Ackerman, 1989).

The variance in the time per error of the different phases can be partially explained by the number of participants in the inspection. The Specification Inspection had six participants, Systems Test Inspection had four participants and Initial Design, Detailed Design and Code inspections had three participants each. It is interesting to note that the time per error where only the three core team members participated is very consistent (Initial Design .3 hrs, Detailed Design .4 hrs, and Code .4 hrs). It seemed that the larger the inspection team, the longer it took to discuss the defective areas. The larger team also had an increased tendency to try to "solve" the problem.

Another factor in the time per error variance is the nature of the product being inspected. The specification inspection generated prolonged discussion of what should be done and how the resulting changes should appear. The code inspections were more absolute; either the code correctly implemented the design, followed good programming practices and adhered to the project standards or it was categorized as an error.

The first CICID release contained a total of 12,326 lines of code (including comments, declarations, etc.). The code size was 46K bytes.

A total of 4.1 major errors per thousand lines of code (50 major errors / 12.326K lines of code) is significantly below Ackerman's reported 7 - 20 major defects per thousand lines of code (Ackerman, 1989). The combined total of 29.1 errors per thousand lines of code (359 errors / 12.326K lines of code) is more than double Buck's reported 8 - 12 defects per thousand lines of newly generated code.

The major errors per thousand lines of code may be lower because the implementation was not extremely difficult. The overall errors per thousand lines of code deviation can probably be attributed to this being the first time the language was used, the first time the process had been applied at Standard Register, and the first time standards had been enforced.

Additional Discussion:

While the importance of limiting the number of participants in the inspection is clearly stated in the IBM literature, the Specification and System Test Plan inspections would not have been effective with only the three core individuals. The marketing representative, department manager and other developer identified concerns during the Specification inspection that would have not been found until later in the process. Having the lead tester involved in the Systems Test inspection also helped to provide a test which was usable by the testing group. Other projects should realize that there will be an increase in the cost per error when there is an increase in the size of the inspection team and should not include inspectors who will not make a substantial contribution to the process.

5.3 INSPECTION PHASE ANALYSIS

5.3.1 SPECIFICATION

Two specification phase inspections were held: CICID Specification and a reinspection of the specification. A total of 64 problems, 17.7% of the total problems found in all inspections, were identified during this phase.

Problem types tracked during the Specification phase were:

	Problem Type	Major	Minor	Total
1.	System Description	3	21	24
2.	Data Requirements	0	1	1
3.	Environmental/Operational Constraints	1	5	6
4.	System Security	0	0	0
5.	Estimate of Next Stages	0	0	0
6.	Reference to Related Materials	0	0	0
7.	User Interface	4	29	33
8.	Other	0	0	0
	TOTAL	8	56	64

Discussion of Results:

The specification was the first formal presentation of the Computer Interfaced Commercial Items Dispenser (CICID) system concepts. No formal requirement document was produced. Rather, impromptu discussions between the marketing representative, department manager and project engineer defined the product's requirements.

Because past projects had been implemented without formal requirements or specifications documents, the specification attempted to combine both the requirements and specification into one document. The high frequency of errors in the System Description and User Interface problem types strongly suggest that a requirements document should have been produced and inspected before the specification phase.

A requirements document and inspection phase could have addressed 25 of the 56 errors found in the specification.

During implementation of the project many changes were made to the specification. (See section 5.4 Specification Change Analysis for discussion of these changes.)

The specification phase and inspections were crucial to the project's success. Many ideas were brought out by the participants which would never have been discovered if there had not been an inspection meeting.

5.3.2 INITIAL DESIGN

One Initial Design inspection was held. The initial design was composed of data flow and state transition diagrams and brief descriptions of the module function and parameters. The inspection identified a total of 20 errors, which was 5.5% of the total errors for all inspections.

Problem types tracked during the Initial Design phase were:

	Problem Type	Major	Minor	Total
1.	System Flow Diagram	0	2	2
2.	Data Dictionary	0	1	1
3.	Data Flow and/or State Transition Diagrams	0	15	15
4.	Module Interface Linkages	0	0	0
5.	Module Description	0	1	1
6.	Performance	0	0	0
7.	Specification Clarification	0	0	0
8.	Specification Incorrect	1	0	1
9.	Standards	0	0	0
10.	Other	0	0	0
	TOTAL	1	19	20

Discussion of Results:

The initial design had very limited descriptions of the modules and parameters. The primary focus was on the data flow and state transition diagrams which attempted to describe the system at the major module level.

More detail should have been included in the initial design. The detailed design and coding phases did not closely follow the initial design. The initial design was primarily a state machine. During detailed design the developers decided that straight-line code with subroutines would be a better implementation than the state machine design.

Even though the straight-line code was a better design for the system, the new systems architecture should have been inspected at the initial design level. The second inspection for the detailed design of the two major functional areas may have been averted with an inspection of the systems architecture changes.

Other changes made to the original initial design were required because of CPU and compiler constraints which required a reorganization of the modules for memory and stack overlaying.

Changes primarily occurred because there was insufficient understanding of how to develop and inspect a diagram based initial design. The training sessions on Yourdon style diagramming did not provide the depth of knowledge which comes from incorporating diagramming techniques into the standard project lifecycle.

The developer's consensus was that the initial design provided an excellent starting point for the evolution of a detailed design. That is, the developers did not believe that their detailed design should be bounded or constrained by the initial design. The experience gained in using the data flow and state transition diagrams on this project however, should help future projects more closely follow the initial design.

The team did update the data flow diagrams to show the final architecture. Everyone agreed that the data flow diagrams were an essential element in the final system documentation.

5.3.3 DETAILED DESIGN

Four Detailed Design inspections including one reinspection were held: Mainline and Pass-through Printing, Exec Entry and Options 1,3,11, Reinspection of Exec Entry and I/O - Check and Voucher Data Structures. A total of 50 errors were identified during this phase, 13.9% of the total errors for all inspections.

Problem types tracked during the Detailed Design phase were:

	Problem Type	Major	Minor	Total
1.	Data Flow Diagram	0	2	2
2.	Data Dictionary	1	13	14
3.	Logic	12	9	21
4.	Data Area Usage	1	1	2
5.	Test and Branch	0	0	0
6.	Return Codes and Messages	0	0	0
7.	Register Usage (ASM only, not used)			
8.	External Linkages	0	0	0
9.	More Detail	0	2	2
10.	Standards	0	0	0
11.	Header or Comments	0	6	6
12.	Initial Design Documentation	0	0	0
13.	Specification	2	0	2
14.	Maintainability	0	0	0
15.	Performance	0	0	0
16.	Other	0	1	1
	TOTAL	16	34	50

Discussion of Results:

Two functional areas and the major data structures were included in the detailed design phase. These areas were chosen because they were not easily implemented from the specification and initial design. Detailed design was selectively performed to keep the development process from being cumbersome. Had all areas required a detailed design, the project would have been considerably lengthened.

Examination of the code inspection data shows that most of the areas did not need a detailed design. All functional areas which did not have a detailed design (except the Console Interface), completed the coding phase with a single inspection.

The console interface code should have had a detailed design. The code was inspected three times with five major and seven minor logic and performance errors which should have been identified in a detailed design.

Functional areas which had a detailed design did not necessarily complete the coding phase without reinspection. Both the Executive Entry and Executive Options 1,3,11 (which had also had two detailed design inspections) and the Mainline and Pass-Through Printer needed two code inspections before they met the exit criteria.

A possible explanation could be that the specification for these areas evolved through the detailed design and coding phases. They required implementation changes because ideas and concepts developed earlier in the project did not work as well as it was thought they would. In the end, this produced a better product than rigidly following the previous phases designs. These were also the most difficult portions of the project.

Inspecting the design of the major data structures separate from a functional area worked very well. It allowed the developers to agree on the structure and usage before they were used in the code. It also provided excellent documentation of these structures.

The extent of detailed design on future projects should be based on a careful examination of the initial design architecture. Only those areas which cannot be easily coded from the specification and initial design documentation should be described in a detailed design. The design of the major data structures should always be inspected before coding begins.

5.3.4 CODE

Ten Code inspections were held including 4 reinspections. Areas inspected were:

- Mainline & Pass-through Printer
- Reinspection Mainline & Pass-through Printer
- Console Interface, Executive Entry and Exec Options 1,2,3
- First Reinspection of Console Interface, Exec Entry and Exec Options 1,2,3
- Second Reinspection of Console Interface
- Exec Options 6,7,8
- Document Parser
- Document Printing
- Areas of Major Change and Problem Fixes
- Signature & Fixed Type Loading

A total of 200 errors, 55.8% of all errors identified in the inspections process were found during the code inspection phase.

Problem types tracked during the Coding phase were:

	Problem Type	Major	Minor	Total
1.	Function Header	1	23	24
2.	Declarations and Defines	2	30	32
3.	Entry and Exit Linkages	0	3	3
4.	Logic	14	47	61
5.	Program Language Usage	1	5	6
6.	Memory Usage	1	0	1
7.	Standards	2	8	10
8.	Performance	2	5	7
9.	Maintainability	0	12	12
10.	Detail Design Error	0	0	0
11.	Comments	0	30	30
12.	Basis Test Paths - Not Used			
13.	Unit Test - Not Used			
14.	Other	0	14	14
	TOTAL	23	177	200

Discussion of Results:

All functional areas had at least one code inspection. Initially, the developers felt there was no need to inspect the code. After several code inspections the developers realized that code inspections had benefits other than finding errors. They provided the opportunity to enforce standards, communicate programming techniques and make certain that program operation was consistent between the two developers.

Analysis of errors in Entry & Exit Linkages, Logic, Program Usage and Memory Usage indicate that these problem types directly contribute to the functionality and correctness of the code.

Errors assigned to Entry & Exit Linkages identified return values which did not match, extra parameters passed to routines and an incorrect pointer which was passed.

The six Program Language Usage and one Memory Usage errors could have caused many problems if they had not been identified during inspection. Errors of these types were expected because neither of the developers had prior PL/M or 8031 processor experience. These errors could have been reduced with language and processor training. Unfortunately, there was neither time or classes available for that language and processor.

Since the primary purpose of the detailed design is to develop the program logic, examination of the code logic errors in modules where there was no detailed design should show which modules were properly excluded from detailed design.

Document Printing and Exec Options 4,9,10,13,14 had four minor and one major logic error which could have been prevented with a detailed design. Exec Options 6,7,and 8 had three detailed design attributable logic errors. Signature and Fixed Type Loading had no attributable logic errors. The Parser had one major and one minor attributable logic error. None of these modules required reinspection. The limited number of logic errors and the absence of reinspections show that these modules would not have significantly benefited from detailed designs.

The Console Interface area, however, was not properly handled with respect to detailed design. The console interface was added to the Executive Entry and Exec Options 1,2 and 3 at the code phase without having had a detailed design. It should have remained separate and undergone a detailed design. The console interface portion accounted for four major detailed design attributable errors and one reinspections with the Exec Entry and Exec Options 1,2 and 3. It also required a reinspection separate from the Exec Entry. Had the the Console Interface been kept separate, the Exec Entry would have had three detailed design attributable logic errors and the the reinspection may not have been required.

Other functional areas which did have a detailed design still had a high frequency of logic errors. Of the 16 logic errors found in Exec Entry & Exec Option 1,2 and 3, and Mainline and Pass-through Printing, 50% should have been found in the detailed design. Some of the errors identified in the code inspections had been correct in the detailed design and were not implemented as designed.

Identifying Standards, Performance and Maintenance errors facilitated a fine-tuning of the modules. Identified were hard coded constants where literals should have been used, subroutines written which duplicated language functions, modules which were not used but remained in the code. Looking specifically for standards violations, performance and maintenance problems resulted in cleaner, more concise and readable programs.

Three problem types, Function Header, Comments and Declaration and Define, would not have hindered the functionality of the code but were important to the overall quality of the code.

Twenty-seven percent of the code errors (54 of 200) were attributed to Function Header and Comments. Declaration and Define errors (32 of 200) identified data items which were duplicates of global variables, defined without being used, required name changes to be readable, needed clarification for better understanding or were not defined consistent with their intended use. While the code would have functioned without these errors having been identified or fixed, it is important for long-term maintenance and use that the program descriptions and data items accurately reflect the code.

Of the 14 Other problem types, 6 were changes to the specification.

5.3.5 SYSTEM TEST PLAN

Two Systems Test inspections were held. The first, Systems Test Part One examined the Mainline program section, Pass-through Printing and Exec Options 1,2,3,5,8. The second, Systems Test Part Two examined Document Printing, and Exec Options 4,9,10,13,14. A total of 25 errors or 6.9% of all errors were identified during the System Testing inspections.

Problem types tracked in the systems test phase were:

	Problem Type	Major	Minor	Total
1.	Approach	0	0	0
2.	Test Description	0	6	6
3.	Test Procedure	2	9	11
4.	System Specification	0	2	2
5.	Hardware/Build Requirements	0	0	0
6.	Operator Instructions	0	6	6
7.	Messages	0	0	0
8.	Other	0	0	0
	TOTAL	2	23	25

Discussion of Results:

This was the first project for which complete system test plans had been written. The inspection's participants included the two developers, the project leader (author of the system test plans) and the lead tester. Including the lead tester was important. He was able to suggest changes to test procedures which would make the systems test easier for the testers to understand.

Inspecting the system test also allowed each individual to be knowledgeable in the test plan. The testers could then ask questions of any inspection participant.

The errors found were in only three problem type categories: Test Description, Test Procedure and Operator Instructions. Since there was no special hardware required, and few error messages which could be displayed, it was expected that the errors would be of these problem types.

The effectiveness of the inspections was demonstrated during the testing process by only two required changes in the test plans. The completeness of the test plans, which the inspection should assure, was demonstrated during beta site testing in which three easily fixed problems were found.

5.4 SPECIFICATION CHANGE ANALYSIS

During the project many changes were made to the specification. Five update releases were made prior to project completion. A detailed description of each change and categorization is included in Appendix B.

For analysis, each change was categorized by the following problem types:

	Problem Type	Major	Minor	Total
1.	Inconsistent With Other Areas	0	3	3
2.	Prompts/Instructions for Moving from Prompt to Prompt	0	4	4
3.	Lister Printing	0	3	3
4.	Omissions	0	8	8
5.	Deletions	0	2	2
6.	Implementation Required Changes	2	7	9
7.	Marketing/Function Required Change	3	6	9
8.	Improved User Interface	2	7	9
9.	Superseded by Later Change	0	1	1
	TOTAL	7	41	48

Discussion of Results:

In a complete project study the evolution of the project may be studied through changes made to the specification. Of primary interest are the events which caused the final product specification to be different than the product specification which satisfied the specification inspection exit criteria.

The dates and content of the revisions parallel the progress of the system. Every three to four weeks from the start of the coding phase to the end of testing (12/8/88 - 2/04/89) there was a revision made to the specification.

Release 1.1 addressed deviations implemented during the coding of the Mainline and Pass-through Printer, Executive Entry & Exec Options 1,2,3,6,7,8. A new Executive Option was added which would display the version information for those machines without a lister printer. In this release there were eighteen minor changes and two major changes; one attributed to implementation required change and the other to a marketing required change. The aggregate minor changes refined the basic system concepts without significantly altering the system described in the specification.

Release 1.2 consisted of three minor changes discovered during reinspection of the code. The limited quantity indicates that Release 1.2 changes should have been held until more items needed changing or the system was ready for release.

Completion of the remaining code inspections precipitated Release 1.3. Three major errors and six minor changes demonstrated a need to update the specification.

Release 1.4 was compiled after most of the modules had been integrated and the user interface could be observed on the machines. Some of the 12 minor and 1 major changes were required to allow the various functions to work consistently; others were made because marketing felt the user interface would be better with the changes.

The final specification revision, Release 1.5 contained 3 minor changes identified during systems testing.

It was beneficial for the specification to always reflect the implementation. Changes in operation or to the user interface had to be communicated to the organization producing the manuals and to marketing. The specification revisions was the most effective method for doing this.

In each of the revisions the fundamental assumptions of the product remained unchanged. Improvements were made to the detail of the implementation producing a consistent, user-friendly system.

The updates to the specification should be collected and then formalized at three project milestone points: all coding completed, integration completed, and system testing completed/project release. Holding all changes until project release may result in manuals which are incorrect. In small projects, changes outside of these points are inefficient in terms of resource consumption.

5.5 SYSTEMS TEST ERROR ANALYSIS

Systems testing was performed by non-technical testers. They performed the systems test and customer simulation test plans and observed problems which are not specifically tested for in either the system or customer simulation test plans.

A total of 30 errors were identified during the systems testing phase. Eleven were major, 19 were minor.

Each error identified in the testing was categorized by problem type as follows:

	Problem Type	Major	Minor	Total
1.	User Interface	2	3	5
2.	Specification Not Correct	1	4	5
3.	Function not as Specified	3	4	7
4.	Problem in Implementation	3	8	11
5.	Unexpected Interaction with External Software Package	2	0	2
6.	Inter-processor Communication	0	0	0
	TOTAL	11	19	30

Discussion of Results:

Testing started January 16, 1989 and concluded March 10, 1989 for a total of 8 weeks of testing. Testing time was 320 hours. The time per error found in testing was 10.7 hours. In comparison, time per error in the inspection process was 0.6 hours.

Others have shown the efficiencies of inspections in relation to testing by examining different statistical items. An operating-system development organization for a large mainframe manufacturer reported that the average effort to find a design defect by inspections was 1.4 hours compared to 8.5 hours to find a defect by testing (Ackerman, 1989). A banking computer-services firm found that it took 4.5 hours to eliminate a defect by unit testing compared to 2.2 hours by inspections (Ackerman, 1989). While these do not show the testing time per error vs the inspection time per error, they do show inspections to be an effective method of finding errors when compared with testing.

There were four test versions (starting with Test 2) provided to the tester during the 8 weeks of testing. (Test 1 was a partially integrated version. By the time a tester was assigned to the project all functions had been integrated into Test 2.)

Over 50% of all test errors (16 of 30) were discovered in Test 2. Eight errors were major, eight errors were minor. As expected, the initial test version showed errors in user interface (specification) and implementation. Instead of immediately fixing the problems and restarting the systems test, all systems tests and some customer simulation tests were completed before installing Test 3. By forcing the project team to examine all of the errors before fixing the problems, the number of test versions was kept to a minimum.

The thoroughness of testing Test 2 contributed to fewer errors identified in Test 3. Nine errors: 2 major and 7 minor were discovered in Test 3. These errors were primarily printing related and would have been difficult to find without the customer simulation testing. Some of the accounts payable packages which we tested with the system were not available prior to Test 3. If they had been available many of the Test 3 problems would have been identified in Test 2.

Version Test 4 had 3 minor errors: a machine timing problem found only because another machine was added to the testing, a slight paper advance problem over a large volume of documents and, an unprintable amount which was lister printed as valid. Finding the first error was a matter of luck in customer simulation testing. The machine that was initially used for testing operated correctly. It wasn't until the second machine was added that the problem was noticed. The second error was a specific test in the systems test which was not run until correct length documents were available. The last error should have been a specific test in the systems test and was caught when the tester entered too many zeros.

Test 5 was intended to be the first release version. A major problem in ending a document session with two previously unavailable accounts payable software packages required changes in how the program recognized the end of a session. The solution may not solve the session ending problem for every possible package of software with which this product could be interfaced. If a session ending method is used which cannot be identified, there will have to be more changes made to the software.

Examination of the errors found in all test versions show that: 23% (7 of the 30 errors) were due to inadequate or incorrect specifications, 43% (13 of the 30 errors) could only have been found in systems test, 27% (8 of the 30 errors) should have been discovered during unit testing and 6% (2 of the 30 errors) should have been discovered in the code inspection.

The errors attributable to lack of formal unit testing are of a small enough quantity to validate the removal of unit tests inspections from the project phases. The time required to formally produce and inspect the unit test would have been significantly more than the time to find the eight errors.

5.6 BETA SITE ERROR ANALYSIS

The true success of inspection process is seen in the Beta Site testing. During 6 weeks of Beta Site testing 3 problems were found. Two of them were major problems in that they affected correct machine operation. One was minor. Each problem should have been found in systems testing but was easily fixed after it was identified.

The impact of inspections on the quality of the Beta release version was immediately observable. The CICID product recorded the lowest error during Beta Test of all Standard Register Products. Marketing and the end customers were pleased with the quality of the product and the ease with which we were able to respond to the three problems.

5.7 COMPARISON WITH A PRIOR PROJECT

A similar project was implemented at Standard Register without the use of inspections. The manual entry Official Items Dispenser release 3 had major areas completely rewritten with two developers and a project leader. A specification and systems test plan were developed. They were not inspected. Development started several months prior to testing and continued during the testing phase.

Testing time and errors were closely tracked. Testing started September 23, 1987 and concluded April 13, 1988 when the first revision of Release 3 was provided to customers. During the 29 weeks, twenty test versions were provided to the testers. A total of 91 errors were identified and fixed. Approximately 1160 hours of testing was expended in this test for a time per error of 12.7 hours.

Many problems were discovered by customers after the product initial release. When the product finally stabilized, five additional versions had been released fixing 12 more errors. A complete list of errors and their categorization is available in appendix B.

Each error identified in the testing was categorized by problem type as follows:

	Problem Type	Major	Minor	Total
1.	User Interface	14	24	38
2.	Specification Not Correct	0	0	0
3.	Function not as Specified	5	7	12
4.	Problem in Implementation	34	14	48
5.	Unexpected Interaction with External Software Package	1	0	1
6.	Inter-processor Communication	4	0	4
	TOTAL	58	45	103

Discussion of Results:

Inspection benefits are obvious when comparing these two projects. The project using inspections was completed and available for beta site customers in 24 weeks, less time than it took to test the non-inspected project. Twenty test versions were provided to the testers as compared with four for the inspected product.

The time to find each error in testing was 10.7 hours for the inspected product and 12.7 hours for the non-inspected product. While this is a 20% reduction in cost, the significant difference can been seen when the time to find errors in inspections is combined with the time spent testing for errors in the inspected product.

In the inspected product 2,211 hours were spent in development and 320 hours in testing. A total of 2531 hours were spent finding 389 errors (30 testing errors and 359 inspection errors). Examining the data in this manner shows 6.5 hours to find an error in the inspected product compared with 12.7 hours for the non-inspected product, nearly one-half the time.

Inspections also kept errors from being amplified in subsequent phases. Using the IBM amplification model (see description in section 2.0), the specification phase could have added an additional 112 errors (.5 * 64 spec errors * 3.5). The design phase could have added 115 additional errors (.33 * 70 design errors * 5). Without inspections, 227 additional errors could have been added.

Other comparisons can be made between the two projects. The non-inspected project has been difficult to maintain and modify while new functions have been easily added to the inspected project. The non-inspected project has minimal comments and documentation. The inspected project followed commenting and documentation standards. Another indication of inspections success is that other project engineers are using inspections.

5.8 OPINIONS OF THE PARTICIPANTS

Central to the success of inspections in the project were the individual developers. Their willing participation in a different method of software development made the implementation of the inspections technique possible. Throughout the project they presented many ideas and changes which helped to mold the large-systems IBM method into a small-systems Standard Register method.

Ming-Reng Chiou is a Software Engineer who started with Standard Register in March of 1988. For one and one-half years prior to working at Standard Register he was a software engineer with Krug International where he used structure charts and data flow diagrams for software design and documentation. His educational background includes a B.S. in Electrical Engineering from Taipei Institute of Technology and an M.S. in Electrical & Computer Engineering from the University of Cincinnati.

When discussing the individual phases and their applicability to software development at Standard Register, Ming believed that a Conceptual Specification (or Requirements) phase needed to be added prior to the Detailed Specification. The Conceptual Specification should include the overall project objective, requirements of the software functions and performance.

The detailed specification document and inspection was the most important aspect of the inspections process. He also indicated that each team member should specifically follow the specification unless there was an agreed upon change made to the specification.

High-level design which partitioned the project into modules and designed the interface between the modules was also an extremely important phase of the project. All team members should be involved in and agree to this design.

Low level design should only partition the modules into major routines and describe what these routine will do and how to interface with these routines. Program logic should not be included in the low level design unless implementation will be in assembly language.

Code inspections do not find many bugs in the programs but are necessary to force the developer to write more readable code. Programmers tend to write better code when they know the other members will be scrutinizing it. Code inspections also help the team members understand each other's implementing algorithms and programming techniques. Finally, code inspections find out bad design approaches.

Unit tests should not be inspected. They take too long to formally write. Unit testing should be performed by the developer and meet all requirements listed in the detailed specification. Systems tests however, should be formally written and inspected.

Overall, Ming felt that the software inspection technique should be followed step-by-step. "A major mistake at the high-level can never be corrected at the lower level. A major mistake at the low-level usually involves only minor change at the higher level design. Therefore, a thorough inspection at each step is the key to successful design and development."

The other developer was Haroon Shah, a Senior Software Engineer with Standard Register since March of 1988. During the preceding year he was a Systems Engineer with Krug International and for two years prior to that was a Software Engineer with H & H Automation. His educational background includes an B.E. in Avionics Engineering from Karachi University in Pakistan, an M.S. in Systems Engineering and Computer Engineering from Wright State University. Haroon had not previously used any software development techniques.

Haroon felt that the specification was vital to the success of any project. An improvement would be a requirements step which would answer the question of what the machine should do before describing how the machine will operate.

The high level design using data flow diagrams was very useful. He emphasized that they are limited in showing process and interrupt procedures. Low level design was useful in the data structures area but not in the module logic.

Code inspections were more important for their side effects than for finding errors. The side effects included:

- All team members knew what they needed to do.
- The person whose code was under scrutiny was more responsible in their coding and commenting practices than if there had been no inspection.
- The project leader knew precisely what each team member was doing and could direct the team easier.
- Team members learned useful programming techniques from inspecting each others code.
- Programming style debates were not useful.

Unit testing should be the responsibility of each individual. Developing unit test plans was not an efficient use of his time. Systems test plans and associated inspections however, were extremely useful and should be formally required department wide.

Haroon felt that "Inspections did help a lot when the two team members software was put together. Fewer problems occurred when the software started working together."

6.0 SUMMARY AND FUTURE DIRECTIONS

This is the only known reported study of inspections in a small systems environment. Use of inspections in this environment required changes to the IBM method in the areas of size of inspection team, roles played by team members, reinspection requirements, Entry and Exit criteria, lifecycle phase inspected and exemption from lifecycle phases where practical.

During the project duration of 24 weeks, data collection and analysis was performed on the results of eighteen inspections, on changes to the specification, on errors found during testing and on errors found in Beta Site. Approximately 10% of the project time was spent preparing for inspections and in the inspection meeting.

A total of 359 errors were found during the inspections, 50 major errors and 309 minor errors. The average time per error found in the inspections was .6 hours. A total of 30 errors were found in systems testing. The average time per error during systems testing was 10.7 hours. Combining the inspection and testing time and errors shows that for the overall process the average time per error was 6.5 hours.

This study also compared the inspected project to a similar non-inspected project. Each project was similar in functionality and was implemented on the same hardware base. The inspected project required 8 weeks to test while the non-inspected project required 29 weeks of testing. A total of 30 testing errors were identified in the inspected project and 103 testing errors in the non-inspected project. The effectiveness of inspections is demonstrated in the inspected project having three errors in beta testing and none in production while the non-inspected project had twelve errors after release requiring five additional production versions.

The most visible indicator of the quality of the process is determined by the errors found in testing and in use. In this study inspections were very effective in facilitating the development of a high quality product.

Less visible indicators of quality are the ease with which problems that are found are fixed, the ease with which another developer assumes responsibility for the product, and the ease with which customer required changes are made. In this project, the three beta site errors were easily found and fixed, the developer who took over the product had no trouble coming up to speed quickly and the additions for specific customers were easily added to the code.

Future projects should have a requirements phase when the project concepts are new or not well understood. After the specification and initial designs have been approved, they should carefully choose the functional areas which will go through detailed design inspections so that the minimum amount of overhead is incurred in the project. All code and system test plans must be inspected.

Updates made to the specification in future projects should be collected and then formalized at three project milestone points: all coding completed, integration completed, and system testing completed/project release. Holding all changes until project release may result in manuals which are incorrect. In small projects, changes outside of these points are inefficient in terms of resource consumption. It would also have been beneficial (because of the quantity of changes to the specification) to reinspect the final specification.

The extent of detailed design on future projects should be based on a careful examination of the initial design architecture. Only those areas which cannot be easily coded from the specification and initial design documentation should be described in a detailed design. The design of the major data structures should always be inspected before coding begins.

In the area of problem type future projects may want to add a new problem type in the coding phase for specification changes.

Assigning categories (missing, wrong, extra) seemed to have limited significance and added little to the analysis of errors on this project. A suggestion for future projects would be to remove category as an error tracking item.

This research suggests additional projects. For example: small systems development using only two developers or a comparison of more than one implementation of small systems development with three developers.

This research demonstrated that the IBM Inspection Technique can be successfully adapted for use in small systems development. Using the inspection concepts of thorough lifecycle product review, defined criteria for proceeding from one phase to the next and error tracking, was instrumental in producing a high quality software product at a reasonable price.

B I B L I O G R A P H Y

1. Ackerman, A. Frank, Fowler, Priscilla J., "Software Inspections and the Industrial Production of Software", in Software Validation, ed H.L. Haunsen, Elservier Science Publishers B.V. (North-Holland) 1984.

2. Ackerman, A. Frank, Buchwald, Lynne S., Lewski, Frank H., "Software Inspections: An Effective Verification Process", IEEE Software, May 1989.

3. Bisant, David B., Lyle, James R., "A Two-Person Inspection Method to Improve Programming Productivity", IEEE Transations on Software Engineering, Vol. 15, No. 10, October 1989.

4. Buck, Robert D., Dobbins, James H., "Application of Software Inspection Methodology in Design and Code", in Software Validation, ed H.L. Haunsen, Elservier Science Publishers B.V. (North-Holland) 1984.

5. Christenson, D.A., Huang, S.T., "A Code Inspection Model for Software Quality Management and Prediction", GLOBECOM '88, IEEE Global Telecommunications Conference and Exhibition - Communications for the Information Age. Conference Record, Vol.1, pp.468-472, 1988.

6. Crossman, Trevor, Some Experiences in the Use of Inspection Teams in Application Development. Standards Bank of South Africa Limited, 1977.

7. Fagan, Michael E., Building Defect-Free Software, Quality Assurance Institution Seminar, Course Notes, 1985.

8. Fagan, Michael E., "Design and Code Inspections to Reduce Errors in Program Development", IBM Systems Journal, Vol. 15, Number 3, 1976.

9. Fagan, Michael E. "Advances in Software Inspections", IEEE Transactions on Software Engineering, Vol. SE-12, No. 7, July 1986.

10. Freedman, Daniel, Weinberg, Gerald, Handbook of Walkthroughs, Inspections and Technical Reviews. Boston: Little, Brown Computer Systems Series, third edition, 1982.

B I B L I O G R A P H Y

11. Graden, Mark W., Horsley, Palma S., "The Effects of Software Inspections on a Major Telecommunications Project", AT&T Technical Journal, May - June 1986, Vol 65, Issue 3, pp. 32 - 40.

12. IBM Corporation. Inspections: Formal Application Walkthroughs. IBM Information System Management Institute, Course Notes, 1987.

13. IBM Corporation. Inspections in Application Development Introduction and Implementation Guidelines. IBM 1977.

14. IBM Corporation. "Structured Walk-throughs: A Project Management Tool", 1973, in Software Design Strategies, second edition., ed. Glenn Bergland, Ronald Gordon, California: IEEE Computer Society Press 1981.

15. Kelly, John C., "Formal Inspection Review Techniques: Experience and Results", Conference Proceedings - Quality Week '89 - Technical Program.

16. Meilir Page-Jones. "Transform Analysis", 1980, in Software Design Strategies, second edition., ed. Glenn Bergland, Ronald Gordon, California: IEEE Computer Society Press 1981.

17. Staton-Reinstein, Rebecca. "Inspections: The Heart of Quality Control", Quality Data Processing Journal of the Quality Assurance Institute, Vol. 3, Number 1, January 1989.

APPENDIX A

FINANCIAL EQUIPMENT DEVELOPMENT
ENGINEERING AND RESEARCH

SOFTWARE PROJECT DEVELOPMENT STANDARD

RELEASE 1.0

MARCH 29, 1989

P. SHERWOOD

TABLE OF CONTENTS

TABLE OF CONTENTS CONT.

PAGES 1 - 12 ARE
CONSIDERED CONFIDENTIAL
AND HAVE BEEN REMOVED

8.0 PROJECT TYPES AND LIFECYCLE PHASE REQUIREMENTS

Projects in Financial Equipment Development generally fall into the following categories: New Products, Significant Changes to Existing Products, Customer Specials and Minor "bug" fixes. These categories require different degrees of rigor to assure their quality.

In general, the product(s) of each phase of development listed for the categories will be inspected. The Inspection technique is described in detail in section 9.0. Section 10.0 describes the phases of development, products produced in each phase, the entry and exit criteria and a checklist of areas to examine when inspecting the product.

8.1 NEW PRODUCT

New product development should included the following phases:

1. Requirements - Optional: may not be required if product requirements are well understood. Phase may include a feasibility study.

2. Specification - Required:

3. Initial Design - Required:

4. Detailed Design - Optional: may not be required when size of project is small and spec/initial design provide enough detail for implementation.

5. Code - Required:

6. Unit Testing - Required: Unit testing is performed by the individual developer. Formal test plans and inspection of the test plan is at the discretion of the project leader. Unit test should consist of a checklist which will exercise each line of code at least once. It should also check boundary conditions and the programs ability to handle erroneous input.

7. Integration Testing - Optional: required only on very large projects in which subsystems need to be tested before inclusion in overall product. Integration testing is generally performed by the individual assigned to perform the integration of the product. This phase should have a checklist of areas to test and may be subject to Inspection at the discretion of the project leader.

8. Systems Test - Required: Systems testing is to be performed by the testing lab. The test plan must be produced and inspected.

8.0 Project Types and Lifecycle Phase Requirements Cont.

8.2 SIGNIFICANT CHANGE TO EXISTING PRODUCT

1. Specification - Required: Only Changes Need Be Detailed.

2. Initial Design - Required: Inspect areas of change and areas impacted by the change.

3. Detailed Design - Optional: may not be required when size of project is small and spec/initial design provide enough detail for implementation.

4. Code - Required: Inspect areas of change.

5. Unit Testing - Required: Unit testing is performed by the individual developer. Formal test plans and inspection of the test plan is at the discretion of the project leader.

6. Systems Test - Required: Inspect new tests. Should call out previous systems tests to make certain areas not included in the change still work.

8.3 CUSTOMER SPECIALS

1. Specification - Required: Inspect and Get Customer Agreement. Only Changes need to be detailed.

2. Initial Design - Optional

3. Code - Required: Inspect if change affects more than 5% of total code.

4. Unit Testing - Required: Unit testing is performed by the individual developer. Formal test plans and inspection of the test plan is at the discretion of the project leader.

5. Systems Test - Required: Inspect new tests. Should call out previous generic systems tests to make certain areas not included in the change still work.

8.4 MINOR CHANGES ("BUG FIX")

1. Software Change Notice - Required if change could affect other projects.

2. Code - Required if change involves more than 5% of base code:

3. Systems Test - Required: Outline only of test to check all changes. May request a standard systems test section if that section will adequately check the fix.

9.0 INSPECTION DESCRIPTION

Software inspections provide a rigorous examination of programming materials. e.g., specifications, design documentation, code listings and test plans.

The results are: project resource savings, improved quality and a method for controlling the software development process.

The inspection process is a set of steps. An inspection done correctly is not just the inspection itself, but the successful following of each step.

INSPECTION PROCESS

PLANNING

OVERVIEW

PREPARATION

INSPECTION MEETING

REWORK

FOLLOW-UP

9.1 INSPECTION STEPS

9.1.1 PLANNING:

- The moderator makes certain the materials meet the input criteria.
- The moderator selects the participants.
- The inspection is scheduled.
- The developer distributes copies of the inspection materials.

9.1.2 OVERVIEW:

- Developer provides an overview of the materials to those who are to participate in the inspection.
- This step may be waved by the moderator if the materials are well known to the participants.

9.1.3 PREPARATION:

- The participants prepare for the inspection by studying the materials. The objective is for all participants to become thoroughly familiar with the inspection materials so that during the inspection meeting they will be better able to find and report errors.
- In preparation the participants will use checklists which suggest areas to look for errors. The checklists have general and project specific elements.
- While examining the materials, they:
  - Check that the materials for the product being inspected match materials from the previous phase(s). For example, detailed design materials should not deviate from the high level design and specification.
  - Understand the required inputs and expected outputs (external linkages to and from each module).
  - Understand the data area environment of each module.
  - Comprehend the control flow and logic.
  - Check that the exit criteria have been met.
  - Note discrepancies or errors found so that they may be recorded during the inspection meeting. Suggestions may be presented and discussed for solutions to problems uncovered. Ultimately the solution must be decided by the developer.

9.1.4 INSPECTION MEETING

The inspection team and the author attend the inspection meeting. Since the meeting is a working session with a fixed objective (to find errors) others are not encouraged to attend.

- Purpose - to find errors, suggest possible solutions, but not recode or rewrite during the meeting.
- Reader paraphrases the material being inspected.
- The group examines the product.
- All errors found are recorded and later assigned an error, type, category and severity.

An inspection should not last more than 2 hours. At the end of the meeting, the moderator goes over the list of errors and the participants agree that the list is correct. (optional)

9.1.5 REWORK

Within one day of the inspection, the moderator distributes the inspection detail report which summarizes the problem list.

- ALL ERRORS FOUND IN THE INSPECTION MUST BE FIXED.
- The developer reports back to the moderator when all problems have been fixed.

9.1.6 FOLLOW-UP

After all errors and ambiguities have been corrected, and if a reinspection has not been scheduled, the moderator verifies that completeness and accuracy of the reworked materials and gives formal approval. This allow the development effort to continue.

- Moderator determines all problems have been fixed and the exit criteria have been met.
- All rework must be completed and verified before moving on to the next phase of the software lifecycle.
- If more than 10% of the inspected product was changed, the materials must be reinspected. A reinspection may also be held if major changes have been made to the fundamental concepts of a functional area or implementation.
- Changes to previous levels are noted (If required).
- Results of the inspection are included in the data base.

9.2 CLASSIFICATION OF ERRORS

9.2.1 Category:

1. Missing
2. Wrong
3. Extra

9.2.2 Severity:

1. Major:
   - Errors which would be difficult to find or fix at a later time.
   - Errors which cause serious problems in operation.
   - Sections left undefined (or not implemented) which are critical to operation.
   - Significant non-mandatory deviations from previous phase documents.

2. Minor:
   - Comments which are missing or not correct.
   - Typographical errors in coding
   - Lessor non-mandatory deviations from previous phase documents.
   - Lessor implementation details which are not correct.
   - Errors which would not be difficult to find or fix at a later time.
   - Errors which cause minor problems in operation.
   - Most operator interface prompting problems.

9.2.3 Type

Each phase has a specific list of types. See the Inspection Report Forms (Appendix D) for detailed list of the types.

10.0 PROJECT LIFECYCLE PHASES

10.1.0 REQUIREMENTS

The requirements identify the scope of the proposed system by defining major functions from a user point of view. Required for a new product concept, it may also be used for major user required changes.

10.1.1 PEOPLE INVOLVED IN INSPECTION:

The requirements should be reviewed by marketing, the department manager, those who will be assigned to design and program the project and others in the group who have worked on similar projects. All issues raised must be addressed before proceeding to the specification phase.

10.1.2 DEFINITION OF THE PRODUCTS PRODUCED:

The Requirements Description:

The requirements description should be limited to 1 - 5 pages. Included in the description are the functional requirements of what needs to be done, the physical requirements of how the system should work, any constraints such as timing and performance needs, and a description of external interfaces.

10.2.0 SPECIFICATION

The specification explains in detail what the system will do to meet the user requirements.

10.2.1 PEOPLE INVOLVED IN INSPECTION

The specification should be reviewed by marketing, the department manager, those who will be assigned to design and program the project and others in the group who have worked on similar projects. Essentially the same group as was involved for the requirements review. This time the specification is inspected. It is required to meet entry and exit criteria. When preparing for the inspection there will be a checklist of items to consider. Before continuing to the next phase all errors must be corrected.

10.2.2 DEFINITION OF THE PRODUCTS PRODUCED:

1. System Description:

   The System Description contains a clear, detailed description of the new system. The narrative should contain sufficient detailed such that no other document would be necessary to determine how the system will work. All user interface elements should be defined.

   If the system is to be implemented in different versions, each version should be described in terms of the functions available.

2. Data Requirements:

   All data requirements of the system identified so far - all files, databases, transactions, report formats, screen displays and other data-oriented system entities - should be described.

3. Environmental and Operational Constraints:

   The environmental and operational constraints can impose performance and geographical requirements, which could determine certain system characteristics. These characteristics are expressed in terms of where and when access to the system is needed, how it is needed and what performance is required. This can include response times, transaction processing rates and similar system attributes.

4. System Security:

   There are three types of controls: sensitive data and processes to be made secure; audit trail requirements; and levels of security. Any or all that apply should be described.

5. Schedule of Next Phase(s): (Optional)

   Include an estimate of the schedules for the High Level Design phase (or next phase, or overall project schedule).

6. Reference to Related Materials:

   Material and documentation generated during the Specification phase or on a previous project which have not been included in the Specification document are referred to in this section.

   Also included in this section are recommendations for the specific program design techniques, language, coding standards and other detailed guidelines to be used in the project.

10.2.3 ENTRY CRITERIA:

1. Complete external description which defines the system from a user or external viewpoint.

2. Data Requirements should be described, if applicable.

3. Environmental and operational constraints should be detailed, if applicable.

4. Security considerations should be described, if applicable.

5. Other items are optional but suggested (Estimate of Cost and Reference to Related Materials).

10.2.4 EXIT CRITERIA:

All entry criteria completed and approved.

10.2.5 CHECK LIST

1. Is the specification consistent with the project (marketing) objective?

2. Is the specification correct and in sufficient detail?

3. Does the specification properly address human factor considerations? For example does it, contain redundant user options, require excessive key strokes or specify dialogue which would not be easily understood.

4. Is anything missing?

5. Is anything included which shouldn't be included?

10.3.0 INITIAL DESIGN

The objective of the Initial Design is to describe how the system will implement the specification. The technical description must be sufficiently detailed so that during the low level design and/or coding phase, all of the system can be completely implemented.

The Initial Design document is intended to be a technical document whose purpose is to provide an unambiguous and sound base with which to start detailed design.

All functional areas in the system are defined in terms of their inputs, outputs, required functions and processes. Timing and performance requirements, where appropriate, are determined for individual modules based on the overall system requirements.

The logical and physical structures of the data for major data items are designed and the system data dictionary is updated with this information.

The initial design is the point at which recommendations can be made for implementing portions of the software with commercially available software packages.

10.3.1 PEOPLE INVOLVED IN INSPECTION

The project leader, developer and all team members should be involved in the initial design. It provides an opportunity to learn the design of the entire system.

10.3.2 DEFINITION OF PRODUCTS PRODUCED:

Data Flow and/or State Transition Diagrams down to conceptual single function modules or states.

Written description of what each transform (DFD Bubble) and/or state will do and possibly how it will do it.

Data Definitions for major data items.

Recommendations for purchased software.

List of changes required to the specification.

10.3.3 ENTRY CRITERIA (Deliverables for Inspection)

1. Data Flow and/or State Transition Diagrams of major modules.
2. Each major module and/or state must have a description of what work will be performed and possibly how it will be performed.
3. Major Data Items defined and documented.
4. Recommendations for purchased software.
5. All changes to the specification since it was approved must be included in the current specification so that the Initial Design and Specification match.

10.3.4 EXIT CRITERIA:
All entry criteria completed and approved.

10.3.5 CHECK LIST

3.5.1 Data Flow and/or State Transition Diagrams

Data Flow:
1. Are all transforms labeled with a description of what is happening to the data?
2. Are asynchronous flows individually listed?
3. Are flows describing different external objects individually listed?
4. Are terminators which play different roles in the system packaged together?
5. Are all inputs to a level from the next higher level accounted for?
6. Are all outputs expected at the next higher level provided?

7. Have error conditions been considered?

State Transition Diagrams:
1. Are all states labeled with their function?
2. Are transactions each triggered by a transaction condition?
3. Are all transaction actions considered?

3.5.2 Data Transform and/or State Descriptions

1. Does the design give a complete and accurate description of the overall function of the module (or state)?

   A. Is all new/changed design areas specified at the functional description level?
   B. Is the design understandable as stated? Can detailed design be implemented from it?
   C. Does the functional description cover all known possible cases? Watch for exception cases.
   D. Are abnormal conditions covered?
   E. For changed functions, is the new logic compatible with the old?

2. Are all required inputs and outputs or transaction conditions and actions defined correctly?

   A. Should an external routine be used rather than performing the function internally?
   B. Is all information flow correctly described?

3. Are performance criteria specified?

   A. Is the function designed optimally for performance and storage?

4. Boundary oversights
   A. Is anything going to fall through the cracks?
   B. Is anything duplicated when it doesn't need to be?
   C. Is each input, function, and output specifically addressed by an identifiable part of the system? Can you prove it?
   D. Are there any misinterpretations of the user interface?

5. Over-adaption
   A. Has any portion of this design received more emphasis than it deserves? What effect has it had?
   B. Is the Design overly constrained.

6. Mistakes
   A. Has anything been forgotten?
   B. What has been done wrong?

7. Sensitivity
   A. Has the User been considered in this design?
   B. Is this design easy to maintain and update?

8. Quality
   A. Is the desired software quality factors included for each major module?
   B. Is the factor appropriate for each major module.

9. Other
   A. Does the design reflect the machine on which it will operator?
   B. Is the design flexible enough to move to other hardware (if this is a future possibility)?
   C. Is the design independent of the programming language that will be used?

3.5.3 All Major Data Items defined and documented.

1. Are all required data definitions specified and defined.

   Are the fields described correctly?
   Have any field definitions been omitted?

2. Is storage criteria specified?

3.5.4. Recommendations For Purchased Software

1. Are benefits and detriments listed?
2. Are the deciding factors correct?
3. Are there other software packages which would better fit the need?

10.4.0 DETAILED DESIGN

The detailed design provides the implementation details required to code a functional area. It takes the transform and or state descriptions developed during the Initial Design and breaks them down into a hierarchy of modules (Structure Charts or list of Modules and subroutines). If needed the logic of all or selected modules may be pseudo-coded.

In some cases the initial design and detailed design may be combined. When sufficient detail exists between the specification and initial design the detailed design phase may be skipped.

10.4.1 PEOPLE INVOLVED IN INSPECTION:

The project leader, developer and select team members should be involved in the detailed level design.

10.4.2 DEFINITION OF PRODUCTS PRODUCED:

The Detailed Design provides:

1. A Structure Chart (or list of modules and subroutines) of each functional area.
2. State Transition Diagrams to the lowest level (if a state machine is the chosen method of implementing the functional area).
3. Data Flow Diagrams documenting to the lowest level.
4. Pseudo-code logic in the range of 3 to 10 Non-Commentary Source Statements (NCSS) per pseudo-code statement. (if needed)

10.4.3 ENTRY CRITERIA:

1. Structure chart (or modules and subroutines list) shows each function to be coded.
2. Data Flow is documented to the lowest level.
3. Pseudo-code represents design to the approximate range of three to ten source code statements per design statement.
4. Pseudo-code is structured following all programming rules.
5. If changes are to existing modules, all new/changed statements should be flagged with release identification.
6. References to data areas should be by variable name.
7. Function calls should specify all required parameters.
8. Parameters to called or invoked routines must be defined. Parameter values passed and return codes expected must be specified.
9. Messages and error messages must be defined.
10. All new/changed major data items must be included in the data dictionary description.
11. Any changes to the Initial Design since it was approved must be updated in the Initial Design so that the Initial design and Detailed design match.

10.4.4 EXIT CRITERIA:

All entry criteria must be completed and documents approved.

10.4.5 CHECK LIST

4.5.1 Structure Chart

4.5.2 Data Flow and/or State Transition Diagrams

Data Flow:
1. Are all transforms labeled with a description of what is happening to the data?
2. Are asynchronous flows individually listed?
3. Are flows describing different eternal objects individually listed?
4. Are terminators which play different roles in the system packaged together?
5. Are all inputs to a level from the next higher level accounted for?
6. Are all outputs expected at the next higher level provided?

7. Have error conditions been considered?

State Transition Diagrams:
1. Are all states labeled with their function?
2. Are transactions each triggered by a transaction condition?
3. Are all transaction actions considered?

4.5.3 Pseudo-Code - Logic

1. Are all constants defined?
2. Are all unique values explicitly tested for input parameters?
3. Are all defaults explicitly checked: for example, blanks in an input message?
4. If character strings are created, are they complete? Are all delimiters shown?
5. If a major data element has many values, are they all checked?
6. Are all counters properly initialized? Are all loops performed correctly?
7. After processing a table entry, should any value be decremented or incremented?
8. Are all routine error conditions adequately defined?
9. Are literals used instead of numbers?
10. When comparing structure elements should all fields be compared?
11. Is the value of a data item used before the item is initialized or has valid information in it?
12. Are all data items shown in the detailed design necessary or are some extraneous?
13. Have the module attributes been specified?
    A. Reenterable?
    B. Library?

4.5.4 Data Area Usage

1. If the module is dependent on creating/adding to/to deleting various areas, are all designated?
2. Are all global data in the system include file?
3. In a structure, does the design show explicitly which field to use?
4. If the program stores into a data area, does it store into the correct field?
5. If a value is fetched from a data area, is the correct field fetched?
6. Should the data area by boundary-aligned?
7. Does a variable have multiple uses? Can conflicts arise?

4.5.5 Test and Branch

1. In a conditional branch are, greater than, equal to, and less than zero tested?
2. After an invocation, should a return code be tested?
3. Are branch paths correct, should true be false (yes be no) and false be true (no be yes)?

4.5.6 Return Codes/Messages

1. Are messages issued for all error conditions?
2. On exits, should a return be set or a message issued?
3. Does the message say what is means?
4. Should more information be supplied in the message?
5. Do return codes in the detailed design match the global definition of the return code as documented?

4.5.7 Register Usage (Assembly Language Only)

1. If a specific register is required, is it specified?
2. Does any macro expansion use a register already in use without saving the data?
3. Is the integrity of all input registers maintained?

4.5.8 External Modules (Linkages)

1. Should a library routine be used rather than what has been designed?
2. Are the parameters passed the right one for the function to be performed?
3. Is the data area mapped as the receiving module expects it to be?
4 Does the processing module set (for output) and process (on input) all required passed parameters? Correctly?

4.5.9 Other

1. Does the detailed design clearly define the modules? Is more detail required to implement?
2. Are any programming standards for the project compromised because of the detailed design?
3. Does the detailed design match the initial design? If not, has the initial design been updated?
4. Does the design impair the performance of this module to any significant degree?

10.5.0 CODE

Code is the language specific implementation of the features and functions determined in the preceding phases. This included declarations, include files, library routines, major modules and their subroutines. The code inspection is designed to assure that the implementation has satisfied all standards and included all aspects described in the preceding phases. With the exception of the Specification Inspection this is the most important inspection.

The inspection must not focus on differing programming styles. The Code inspection should only review the code for violations in programming standards, incompatibility with other code, and logic and implementation errors.

Each routine will be paraphrased roughly line for line (including the header and declarations). Library and include files must also be reviewed in the appropriate inspection.

10.5.1 PEOPLE INVOLVED IN THE INSPECTION

The Project Leader, Developer and at most two others should be involved in the inspection of the code. The others involved should be directly impacted by the code. e.g. person who takes the input or uses the output in their modules.

10.5.2 DEFINITION OF PRODUCTS PRODUCED:

The deliverables for inspection are:

1. Source code listing, including the cross reference, before the major testing or the unit testing has been performed.

   Code must follow the appropriate standards.

10.5.3 ENTRY CRITERIA:

1. Each module has a complete file header as detailed in the Programming Standard. Subroutines/Procedures must have a procedure header.
2. Source code listings with cross listing which has no compiler errors.

10.5.4 EXIT CRITERIA:

1. The code does not violate the programming standard. Only exception is when a standard is waived by the inspection team.
2. Function/Procedure headers are complete and up-to-date.
3. Changes to the design or specification determined during implementation have been made to the appropriate document so that the design and code matches.

10.5.5 CHECK LIST:

Check lists for code are language specific. See the appropriate language checklist when preparing for a code inspection.

General Code Check List

1. Function
   - Is the function clearly expressed?
   - Will the function perform reliably?

2. Form:
   - Is the programming style clean, clear upon examination of the whole program.
   - Can it be understood by reviewers of all skill levels?
   - Are there areas which should be made into subroutines due to repeated code segments?
   - Are the comments useful or are they simply alibis for poor coding?
   - Is the level of detail consistent?
   - Are standards followed?
   - Is initialization properly performed? Does the routine clean up after itself?

3. Economy:
   - Are there redundant operations which provide no benefit?
   - Is storage used consistent?
   - Will the module be costly to modify?
   - Is the implementation done in the simplest possible manner?

10.6.0 UNIT TEST PLAN

The unit test plan (if required) should be inspected at the same time as the code for the required functional area. A unit test will examine all independent paths through the software using the basis test method. In addition it will include boundary conditions and unexpected inputs not covered by the basis test set. This test should be in the form of a checklist of inputs to the program. It may be run under emulation if desired.

10.6.1 PEOPLE INVOLVED IN INSPECTION

The Project Leader, Developer and at most two others should be involved in the inspection of the code and unit test. The others involved should be directly impacted by the code. e.g. person who takes the input or uses the output in their modules.

The Unit Test will be examined case by case to determine if it will test the code.

10.6.2 DEFINITION OF PRODUCTS PRODUCED

The deliverables for inspection are:

1. Plan for testing the code being reviewed. This should include the Basis Set of test cases and other test cases which will be required to determine the function is working correctly. The test cases chosen must check the boundary conditions. Additionally, discontinuities must be tested. (e.g. rollover from month to month, year to year and how critical functions react to these conditions.)

10.6.3 ENTRY CRITERIA

Unit test plan which minimally tests the software.

10.6.4 EXIT CRITERIA

Unit test adequately tests the function.

10.6.5 CHECK LIST

1. Test Cases:

    1. Is each path through major modules tested?
    2. Does the path through major modules test the submodules adequately? If not, are special test cases provided for the submodules?
    3. Are unusual combinations considered?
    4. Are test cases missing?
    5. Are there redundant cases?

10.7.0 SYSTEM TEST PLAN

The primary objective of the system test is to provide assurance that the system operates correctly within its intended environment.

The system test plan is produced from the system specification. In contrast to the unit test plan, the systems test plan is written with no direct knowledge of implementation. The purpose of the system test plan is to test the system from the users point of view.

The system test plan may be produced in sections which follow the functional divisions of the system. It is best to partition the inspections into material which can be inspection in 2 hours or less.

10.7.1 PEOPLE INVOLVED IN INSPECTION

The project leader, test plan developer, lead tester and individual(s)who produced the areas to be tested should attend the system test plan inspection.

10.7.2 DEFINITION OF PRODUCTS PRODUCED

1. Overall systems test. May be broken into sections which follow the functional area of the system.
2. User simulation section which for a period of time operates the software as it is intended or expected to be used.

10.7.3 ENTRY CRITERIA

1. Test must cover each section of the system or functional area.
2. Tests must be from a user(operator) point of view.
3. Routine tests must be performed without emulation.
4. Error conditions may be tested with emulation if difficult/impossible to cause the error condition.
5. All areas of the test plan must be complete.
6. Input and output boundary conditions must be tested.

10.7.4 EXIT CRITERIA

All entrance criteria are satisfied.

10.7.5 CHECK LIST

1. Are tests sufficient to provide confidence that the function being tested operates correctly?
2. Is the testing approach feasible?
3. Are all now/changed user interactions exercised?
4. Is a sufficient number of defaults exercised?
5. Are messages verified?
6. Are error paths exercised?
7. Are sufficient and proper tests identified to verify previously tested functions?
8. Are there simulator and hardware dependencies that are not addressed?
9. Are there any outstanding design changes that will invalidate the completeness of the test plan?
10. Are input and output boundary conditions included, complete?
11. After formal testing has time been specifically designated for attempt to break the system.

10.7.6 SYSTEMS TESTING PROCEDURE

The project leader is responsible for the quality of the the testing.

1. The rigor of the testing must be monitored by the Project Leader. (How well are the testers doing their job?)
2. Testing logs should be reviewed 1 or 2 times each week to record errors found in testing and demonstrate the importance of the logs to the testers.
3. Error lists of all problems found in testing must be maintained for each project. If more than one person is working on the project the error list must be distributed to those working on the project. All errors must be fixed or addressed before releasing a version of software.
4. Versions of software provided to the testers should included a written list of errors fixed, areas changed, and any specific tests or areas they must check out. All fixed, areas changed etc., should be checked out before continuing with the testing.

11.0 PROJECT LEADER CHECK LISTS

11.1 DEVELOPMENT CHECK LIST

PROJECT:_____

PROJECT TYPE: NEW PRODUCT MAJ CHANGE MINOR CHANGE CUSTOMER SPECIAL
(circle one)

DATE STARTED:_____

DATE COMPLETED	ITEM
_____	1. Requirements - Optional: Requirements Inspected _____ (Y/N) If No, Why. _____
_____	2. Specification - Required: Specification Inspected _____ (Y/N) If No, Why. _____
_____	3. Initial Design - Required: Initial Design Inspected _____ (Y/N) If No, Why. _____
_____	4. Detailed Design - Optional: Detailed Design Inspected _____ (Y/N) If No, Why. _____
_____	5. Code - Required: Number of Functional Areas No._____ Code for Each Area Inspected _____ (Y/N) If No, Why. _____
_____	6. Unit Testing - Required: Unit Test for Each Functional Area _____ (Y/N) If No, Why. _____ Unit Test for Each Area Inspected _____(Y/N) If No, Why. _____
_____	7. Integration Testing - Optional:
_____	8. Systems Test - Required: Systems Test Inspected _____ (Y/N) If No, Why. _____
_____	9. Software Change Notice - Required for Minor Fixes Which Impact Other Projects.

11.0 Project Leader Check Lists Cont.

11.2 INSPECTION CHECK LIST

AREA INSPECTED:_____

INSPECTION TYPE:_____

Date Completed	Item
_____	1. PLANNING:
_____	Input Criteria Met.
_____	Participants Selected
_____	Inspection Scheduled.
_____	Inspection Materials Distributed (by author).
_____	2. OVERVIEW:
_____	3. PREPARATION: Total Time Spent in Preparation _____hr
_____	4. INSPECTION MEETING
	Length of Inspection _____hr
	Number of Participants _____
	Total Inspection Time _____hr
	Total of Major Errors _____
	Total of Minor Errors _____
	Total all Errors _____
_____	5. REWORK
_____	6. FOLLOW-UP
	Reinspection Required? _____ (Y/N)
	All errors fixed? _____ (Y/N)
	Exit Criteria Met _____ (Y/N)

11.3 RELEASE CHECK LIST

PROJECT:_____

Date Completed	Item
_____	1. Engineering Release Form
_____	2. Software Number and ECN. No._____
_____	3. Programmed Media No. & ECN. No._____
_____	4. Prom Pack Assembly No.& ECN. No._____
_____	5. Software Kit No. and ECN. No._____
_____	6. Other _____
_____	7. Binder
_____	Diskette(s)
_____	Source Listings
_____	Documentation
_____	8. Beta Site Successfully Completed/Customer Accept Software
_____	9. Release Complete.

APPENDIX B

Specification Inspection for:
Computer Interfaced Commercial Items Dispenser
Held 9/19/88
Total Prep Time 23 hrs/6 participants
Total Inspection Time 5 hrs/ 6 participants

Inspection Type	Pr Type	Severity	Category		Description
spec	1	Minor	M	p1	Expand general description.
spec	3	Minor	W	p1	Real time clock will be in machine. Time information is required for start/stop time.
spec	1	Major	W	p1	Rework on line and off line designation to reflect always on line unless manually taken off line.
spec	1	Minor	W	p2	When going off line manually always complete the current document.
spec	7	Minor	E	p3	Rework status showing errors only.
spec	1	Minor	W	p5	There are 2 WAYS to initiate modem.
spec	1	Minor	W	p5	Rework modem/rs232 to reflect only manually off line.
spec	1	Major	W	p6	When buffer is 90% full stop receiving and start receiving when ( ) percent full.
spec	1	Minor	E	p6	Remove paper check when moving to new line.
spec	7	Minor	W	p6	No paper prompt should use RETURN.
spec	3	Minor	W	p6	Pass through printer is OKI 192. Other printers in non-graphic mode may be supported in the future.
spec	7	Minor	E	p7	Exec pass code is 6 chars not 8.
spec	7	Minor	W	p7	Default to no exec code required.
spec	7	Minor	M	p7	Should have two level executive.
spec	7	Minor	M	p8	Clear print buffer is always N. Use description from print sample doc.
spec	1	Minor	W	p9	Clear all memory is option 3.
spec	7	Minor	W	p10	In print doc grid option prompt use LOAD not INSERT.
spec	7	Minor	W	p10	Pass through printer session id should be 15 characters.
spec	1	Major	W	p11	Rework Document Configuration.
spec	7	Minor	E	p14	Sample document should be filled in with x's.
spec	7	Major	W	p15	Modem or RS232 not both.
spec	7	Minor	E	p15	Assume port 1/modem, 2/RS232.
spec	7	Minor	E	p16	No need for modem auto answer session type.
spec	7	Major	M	gen	Provide for printer functions TOF, FF, LF.
spec	1	Minor	M	gen	Modem baud should be s/w selection.
spec	1	Minor	M	gen	Add communication specification.

Specification Inspection for:
Computer Interfaced Commercial Items Dispenser: Draft 2
Held 10/5/88
Total Prep Time 19.5 hrs/6 participants
Total Inspection Time 3.5 hrs/ 7 participants

Inspection Type	Pr Type	Severity	Category		Description
spec	1	Minor	M	g	Modem Baud Rate should be software selectable.
spec	1	Minor	M	p1	Print a journal list if configured.
spec	1	Minor	M	p2	Also, new session will not be allowed until buffer has been printed.
spec	7	Minor	M	p3	Indicate No Paper has priority over Docs to Print.
spec	2	Minor	W	p3	Session Id's will be 20 characters.
spec	7	Minor	W	p4	If type changed with docs to print, prompt CANNOT change session etc.
spec	3	Minor	M	p5	In Modem descr. clear buffer if incomplete transmission.
spec	1	Minor	W	p5	RS232 description has 2 number 1's.
spec	1	Minor	W	p5	References to CID not OID.
spec	3	Minor	M	p6	Future Enhancements include other than XON/XOFF.
spec	1	Minor	M	p7	Data Reception Descr RS232 only.
spec	3	Minor	M	p7	OKI or Epson will be supported.
spec	7	Minor	W	p8	Pass through printer prompt for online - PRESS CNTRL FOR OFFLINE/ESC TO STOP if ESC prompt ESC-START T-TOF F-FF L-LF N-NPR C-CPR
spec	7	Major	M	p9	Exec entry must consider 2nd exec and no access if option is invalid.
spec	7	Minor	M	p9	Scroll through options listing name of option for ease of use.
spec	7	Minor	M	p1	Should be: change MASTER exec code.
spec	7	Minor	M	p9	Option 1 remove reference to up/down arrows for lines 1 - 5.
spec	7	Minor	M	p10	Prompt for less than 6 chars needs a RETURN instead of CLEAR.
spec	1	Minor	M	p10	Remove second exec report.
spec	7	Minor	M	p11	Scroll through options for 2nd exec. An ESC is required to exit.
spec	1	Minor	W	p12	Exec default pass code is 1 - 6.
spec	1	Minor	M	p13	Print Grid will Print Double.
spec	7	Minor	W	p13	In Print Grid - go back to SELECT OPTION when printing starts.
spec	1	Minor	M	p13	Expand general description.
spec	7	Minor	W	p13	Default pass through printer session type id to PRINTER.

Specification Inspection for:
Computer Interfaced Commercial Items Dispenser: Draft 2 Cont.

Inspection Type	Pr Type	Severity	Category		Description
spec	7	Minor	E	p14	Lines per page should be deleted.
spec	7	Major	E	p15	Rework Input and Output so that only Executive definable types take up the slots.
spec	7	Minor	W	p15	Input (and Output) types will have a default session type of NOT USED 1 - NOT USED maximum type number.
spec	1	Major	W	p15	Set up Input type so that data may be found in both the check and voucher areas.
spec	7	Minor	W	p15	Use voucher top and bottom for consistency.
spec	7	Minor	E	p16	Remove accept changed from Input and Output session definitions.
spec	7	Minor	E	p16	Remove Y/N from Input and Output session definitions.
spec	7	Minor	E	p19	Remove WILL from line 9 prompt.
spec	1	Minor	M	p21	Provide order of displaying RAM and ROM based Input and Output Types.
spec	7	Minor	M	p22	Turn off Session Type if an Input or Output Type is not chosen.
spec	7	Minor	M	p22	Default Session ID's are 1 - 4.
spec	1	Minor	W	p23	Print Sample document is option 9 not option 10.
spec	1	Minor	M	p1	Add executive option for input printer determination.

Initial Design Inspection for:
Computer Interfaced Commercial Items Dispenser
Held 10/10/88
Total Prep Time 2 hrs/2 participants
Total Inspection Time 6 hrs/ 3 participants

Inspection Type	Pr Type	Severity	Category	Description
				Context Diagram
hld	1	Minor	W	ESC and CNTRL go with the operator not the external device.
hld	1	Minor	W	Signalling goes both directions from External device, OKI printer and Lister printer.
				Level 1 DFD
hld	3	Minor	M	Exe writes to the clock.
hld	2	Minor	M	Rename Print Data to Input Data.
hld	3	Minor	M	Exec acts on input data to clear buffer.
hld	3	Minor	M	Data Reception interrogates config info.
hld	3	Minor	M	Off line uses config info.
				Level 2 - DFD 2 On line
hld	5	Minor	M	Rename session type module to Session Type Inquire.
hld	3	Minor	M	Session Type invokes pass through.
hld	3	Minor	M	Doc Printing sends out Print Data and Journal Printing.
hld	3	Minor	M	Pass through sends out Print Data and Journal Printing.
				Level 2 - STD 2 On Line
hld	3	Minor	M	Session Type transition conditions are missing.
hld	8	Major	W	Specification change - no paper doesn't abort, CNTRL is only way out.
				Level 3 - DFD 2.2 Doc Printing
hld	3	Minor	M	Move check for CNTRL or ESC to parse input module.
hld	3	Minor	M	Lister will query config info.
hld	3	Minor	M	Print will receive signalling info.
hld	3	Minor	M	Lister will receive signalling info.
				Level 3 - STD 2.2 Doc Printing
hld	3	Minor	M	Move CNTRL to Parse Input.
hld	3	Minor	M	States need to be numbered.
				Level 2 - DFD 4 Exec
hld	3	Minor	M	Add input data.

Detailed Design Inspection for: Power-up, Movement Between States and Pass-through Printer CICID
Held 10/12/88
Total Prep Time 2 hrs/2 participants
Total Inspection Time 6 hrs/ 3 participants

Inspection Type	Pr Type	Severity	Category		Description
dd	3	Major	M	p1	Add StateVar = PowerUp before beginning power up activities.
dd	3	Minor	M	p1	Add initialize global variables to initialing timers.
dd	16	Minor	W	p1	Get valid date and time instead of prompt for them.
dd	3	Major	M	p1	Detailed Description of on line does not say when printing is done.
dd	9	Minor	M	p1	Expand error condition checks description.
dd	1	Minor	M	p2	Expand Data Flow Diagrams to show routines called in pass through printing.
dd	1	Minor	M	p2	RS232 ISR DFD will need to access the StateVar.
dd	11	Minor	W	p3	Data will be thrown away when in other than on line state.
dd	11	Minor	E	p3	Delete disabling interrupts as an alternative to discarding chars.
dd	2	Minor	E	p3	In Variable description remove reference to port 2.
dd	2	Minor	M	p3	In Variable description add vars for read pointer and insert pointer for the input data buffer.
dd	2	Major	M	p3	In Variable description add description of input data buffer.
dd	2	Minor	W	p3	In Variable descr rework description of the high and low threshold flags.
dd	2	Minor	M	p3	In Variable description StateVar must include reference to states other than on and off line.
dd	3	Major	W	p4	RS232 ISR, if StateVar not = OnLine
dd	3	Major	W	p4	Incount must be incremented when a character is received into buffer.
dd	3	Minor	M	p4	Need to return from interrupt after inserting a character in the Input Data buffer.
dd	3	Minor	M	p4	In Pass-thru, housekeeping should include sending XON at first entry.
dd	9	Minor	M	p5	Expand rectify errors.
dd	3	Minor	W	p5	Do not ask operator to continue printing if a CONTROL is received.
dd	3	Major	M	p5	When character is removed from Input buffer, InCount must be decremented.

Detailed Design Inspection for: CICID Executive Entry, options 1, 3, & 11
Held 10/17/88
Total Prep Time 1.5 hrs/2 participants
Total Inspection Time 1.5 hrs/ 3 participants
Reinspection Required? Y

Inspection Type	Pr Type	Severity	Category		Description
dd	13	Major	W	p1	Executive code will default to required so that clearing or init pack will work correctly.
dd	2	Minor	W	p1	Use literals (constants) for the number of options and other changeable size parameters.
dd	2	Minor	W	p1	The type of Exec2nd_opt should be boolean to show how it is used.
dd	3	Minor	W	p1	When an option entered is out of range redisplay the select option prompt.
dd	3	Major	W	p1	Rework exec entry to match the Initial Design.
dd	3	Major	W	p2	In Set Exec code display the current value for exec code required.
dd	3	Major	M	p2	Must check for Valid exec code.
dd	3	Minor	E	p2	Do not call editor for second exec until the prompt is displayed.
dd	3	Major	M	p2	Set all exec2nd_opt to Y if all valid for 2nd exec is chosen.
dd	13	Major	W	p2	To exit 2nd exec option selection an ESC is required.
dd	3	Major	M	p3	Clearing memory reboots and resets the necessary parameters.

Detailed Design Reinspection for: CICID Executive Entry, options 1, 3, 11.
Held 10/19/88
Total Prep Time 1 hrs/2 participants
Total Inspection Time 4.5 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category		Description
dd	11	Minor	M	p1	Expand comments to show true meaning of nulls in the first byte of the master and second exec codes
dd	4	Minor	M	p1	Add BOOLEAN lit BYTE to declares.
dd	2	Minor	W	p1	All references to exec code should be master exec code.
dd	3	Major	M	p2	When displaying option selections must include the descr of the option.
dd	3	Major	M	p2	In set exec code check for ESC at each prompt.
dd	3	Minor	W	p3	Second exec option entry requires an up or down arrow or RETURN.
dd	3	Minor	W	p3	In exec2nd_option i=0 not i=1 when wrapping around.
dd	3	Minor	E	p3	Remove temp option array in exec2nd_option.

Detailed Design Inspection for: Input, Output, Check and Voucher Data Structures
Held 11/15/88
Total Prep Time 1 hrs/2 participants
Total Inspection Time 4.5 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category		Description
dd	4	Major	W	p1	Use a value > max session to distinguish no session selected. Then use Pass through = 0, Doc sessions = 1 - 4 and max session is lit '5'.
dd	11	Minor	M	p1	Show examples for input type using 0 array index, so that 0 - 9 will be evident for input types 1 - 10.
dd	11	Minor	M	p1	Add to comments what format the date will take if DateMMDDYY is FALSE.
dd	11	Minor	M	p1	In general comment for LIC, indicate that all should be zero if the field does not exist.
dd	2	Minor	M	p1	Missing Date variable for InTypes structure.
dd	2	Minor	M	p2	Missing VoucherOnForm for OutType structure.
dd	2	Minor	M	p2	Add label to OpAgent and EmplVenBr OutTypes structure.
dd	2	Minor	M	p2	Amounts should be ASCII 12 bytes or BCD 6 bytes.
dd	2	Minor	E	p2	Delete CheckDefaults variable.
dd	4	Minor	W	p3	Voucher should be a simple array and used as a list.
dd	2	Minor	M	p3	CheckInfo is missing payee array.

Code Inspection for: Mainline and Pass-through Printer
Held 11/8/88
Total Prep Time 3 hrs/3 participants
Total Inspection Time 7.5 hrs/ 3 participants
Reinspection Required? Y

Inspection Type	Pr Type	Severity	Category	Description
c&ut	7	Minor	M	Variables should follow upper and lower case consistently throughout the code.
c&ut	2	Minor	E	Global.inc - remove the add literal it is a duplicate.
c&ut	2	Minor	E	Keydef.inc - remove OTHER3_KEY it is a duplicate of ESC.
c&ut	2	Minor	E	Keydef.inc - remove CTRL and change CONTROL_KEY to CTRL_KEY
c&ut	2	Minor	E	Change ciphardw.inc to hardware.inc. POWERUP.PLM
c&ut	11	Minor	W	Comments for comm1Active andcomm2Active have modem and RS232 mixed up.
c&ut	4	Minor	W	SessionSelection = Passthrough should be removed from the power up sequence. SessionSelection needs to be what it was last time a session was run.
c&ut	4	Major	M	Must establish time and date upon power-up, through realtime clock or op entry.
c&ut	4	Major	M	Timers 0 & 1 interrupts must be defined.
c&ut	14	Minor	M	Several prompts are missing the slash.
c&ut	14	Minor	W	Mainline-docs left status should be DOCS TO PRINT.
c&ut	7	Minor	W	Use a constant NULL_KEY instead of a number for key = 0ffh.
c&ut	9	Minor	W	Change OnLineProc to ProceedToOnline and ExecProc to ProceedToExec.
c&ut	9	Minor	W	Change selection to NewSelection.
c&ut	9	Minor	W	Set (and Reset) the state variable in ProceedToOnline.
c&ut	4	Major	E	Remove extra lines in MenuOffToOnline procedure.
c&ut	4	Major	M	After loading paper must release the printer.
c&ut	14	Minor	E	Remove ':' in select session prompt. PRT_COMM.PLM
c&ut	9	Minor	W	Separate printer and RS232 routines and interrupt service routines.
c&ut	14	Minor	E	In SendSomeMore use constant instead of binary bit mask.
c&ut	2	Minor	W	Change CommHoldOff to CommHold.
c&ut	4	Major	W	In passthru printer ch = 0FF not 0ch.
c&ut	13	Major	M	Unit Test must be provided with completed mainline/printer portion.

Code Reinspection for: Mainline/Pass Through Printer and Interrupts
Held 11/29/88
Total Prep Time 2 hrs/2 participants
Total Inspection Time 3 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category	Description
c&ut	14	Minor	M	In pass_prt, Each occurrence of the following prompt is missing a slash. (PRESS ... /S TO STOP).
c&ut	1	Minor	W	STDLIB2 filename is incorrect in header.
c&ut	2	Minor	W	All constants should be in upper case.
c&ut	5	Minor	W	In power-up do not use FFH for assignment use a constant literally defined as FFH.
c&ut	4	Minor	W	Examine interrupts being disabled while initialization occurs. Move lines 410-413 to power-up init.
c&ut	5	Minor	W	pass_prt, p4 comm.incount and comm.Incount need to be consistent in use of capitol letters.
c&ut	14	Minor	W	All Y/N prompts need two blanks after item to answer Y/N and one blank before cursor position. e.g. MODEM Y/N Y
c&ut	4	Minor	W	In STDLIB destructive edit, delete setting of the cursor. curpos = i + start pos
c&ut	4	Minor	W	In exec p5, line 260 should be if answer = YES not key = YES.
c&ut	13	Not Count		We decided to eliminate the unit test per say and perform unit testing with the specification and a log of the testing performed.

Code Inspection for: Console Interface, Executive Entry & Options 1,2,3
Held 11/9/88
Total Prep Time 3 hrs/2 participants
Total Inspection Time 9 hrs/ 3 participants
Reinspection Required? Y

Inspection Type	Pr Type	Severity	Category	Description
c&ut	13	Major	M	Unit Test Plan was not included.
c&ut	11	Minor	M	Comments for console variables need to be expanded
c&ut	2	Minor	E	xmit_handler does not use var xmit_char.
c&ut	4	Major	W	rcv_handler must be reworked to remove do while xmit_done loop.
c&ut	9	Minor	E	Proc wait is not needed use PLM funct.
c&ut	11	Minor	M	Comment kbhit so that it is understood that this routine must be checked before calling getkey since getkey does not return until a key is found.
c&ut	11	Minor	M	Comments for the queue should say that if the queue size ever exceeds 256 bytes the pointers will have to be checked for wrap around.
c&ut	8	Minor	E	Delete getsr routine it is not needed.
c&ut	2	Minor	W	Use another literal instead of EOT for keyboard type. EOT has other communications connotations.
c&ut	11	Minor	M	Comment key packet and lister modules.
c&ut	4	Major	M	Getkey needs to send NAK for incorrect checksum. It also needs to send NAK if it doesn't get EOL.
c&ut	11	Minor	M	Comments for display_code and display_data should say string terminated with an EOL (not CR). Also include info on cursor.
c&ut	4	Major	M	Must be able to disable cursor after displaying a string.
c&ut	2	Minor	M	Use addr for pointers instead of word.
c&ut	2	Minor	E	Remove status and dummy from lister_state.
&ut	2	Minor	W	Use different literal than STX to indicate start of lister messages.
c&ut	4	Minor	E	In clr_display don't bother clearing the first 8 spaces.
c&ut	3	Minor	W	strcmp - the comments and code do not match for return values.
c&ut	4	Major	W	Rework edit_hidden_str and edit_xmask_string.
c&ut	2	Minor	W	Master executive pass code must be absolutely located.
c&ut	4	Minor	W	When displaying options choices for 2nd exec do not display the # of the option.
c&ut	2	Minor	E	Expand general description.

Code Inspection for: Console Interface, Exec Entry & Options 1,2,3 Cont.

Inspection Type	Pr Type	Severity	Category	Description
c&ut	4	Minor	M	2nd Exec must have option 0 set to true to exit the executive mode.
c&ut	4	Minor	W	set_execode - master_execcode(0) <>NULL should be second _execcode.
c&ut	5	Minor	W	When looking for Y or N entries use YES or NO instead of TRUE or FALSE to determine entry type.
c&ut	4	Minor	M	Before checking if all options are valid for second exec, set i=0.
c&ut	4	Major	M	clear_all_mem opt 3 must reboot when clearing is complete.
c&ut	2	Minor	W	Rename exec_mode to proceed_to_exec so that it will match higher level.
c&ut	4	Minor	M	When entering exec mode indicate master or second exec on lister.
c&ut	14	Minor	W	SPEC CHANGE:Select Option does not allow a numeric entry, just arrow around to the option desired and press RETURN.
c&ut	4	Minor	M	Before lister printing anything, check to see that the lister printer has been turned on.

Code Reinspection for: Console Interface, Exec Entry and Options 1,2,3
Held 11/23/88
Total Prep Time 3 hrs/2 participants
Total Inspection Time 4.5 hrs/ 3 participants
Reinspection Required? Y - Console Interface Redesign Only

Inspection Type	Pr Type	Severity	Category	Description
c&ut	8	Major	M	Go back to detailed design for console interface. Address issues: Need to retransmit message when a NAK is received, rcv_handler should not store a NAK, the ACK should be sent from the rcv_handler not from the application level.
c&ut	4	Minor	W	get_key should not return a null.
c&ut	11	Minor	M	Provide comments on how the cursor works. Include: Turn on/off, movement, etc.
c&ut	4	Minor	W	Move ack_need = TRUE just prior to call putchar(chk_sum).
c&ut	4	Minor	W	Check if lister configured in low level lister subroutines.
c&ut	2	Minor	W	Change sum to bin_num and min to digit_count in itoa.
c&ut	11	Minor	M	Comment routine itoa so non-C programmers can understand module.
c&ut	11	Minor	M	Add display_position to comments about time literals.
c&ut	4	Minor	M	Make more distinction between the real-time clock being updated and the software clock being updated.
c&ut	2	Minor	E	Remove curpos and tmp_str from exec.
c&ut	4	Minor	M	If no exec required must still lister print EXECUTIVE MODE upon entry to exec mode.
c&ut	9	Minor	W	00 is a valid year i.e. year 2000.
c&ut	13	Major	W	The unit test is too detailed. The unit test should be a list of things to check out.

Code Reinspection for: Console Interface
Held 11/30/88
Total Prep Time 3 hrs/2 participants
Total Inspection Time 1.5 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category	Description
c&ut	11	Minor	M	rcv_handler, line 75, comment this line to indicate that you are looking for the checksum.
c&ut	2	Minor	W	getkey - use key instead of rcv_char to better describe the use of the variable.
c&ut	2	Minor	E	getkey - chk_sum is not used remove from declarations.
c&ut	11	Minor	M	Provide application instructions for working with now. i.e. each time the current time is required disable timer 1 interrupt copy now to a temporary variable and then reenable the interrupt.
c&ut	4	Minor	W	Time p 8 - when counting count down for better results.

Code Inspection for: Executive Options 6, 7, 8.
Held 12/13/88
Total Prep Time 4.5 hrs/2 participants
Total Inspection Time 6 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category	Description
code	1	Minor	W	zdatastr.inc name does not need the z.
code	11	Minor	M	zdatastr - comments for intype/outtype literals should include when the lits must be updated.
code	1	Major	M	Explain the strategy behind the fixed types for intypes and outtype arrays.
code	11	Minor	W	zdatastr.inc - delete comment for intype/outtypes in session types in which only user definable indexes are used.
code	7	Major	W	Use library functions wherever possible to eliminate duplication of code and lessen the amount of testing required.
code	2	Minor	W	EXEC22 - CodeToMemCopy Size change var name BytesToMov to ByteCount.
code	6	Major	W	Throughout exec22.plm do not use internal variables, they should be aux.
code	9	Minor	W	All printer commands e.g. SetTOF should reside in the printer driver module.
code	4	Minor	W	GetBin_CR_ESC eliminate the use of the left and right arrows in numeric entry.
code	4	Minor	M	GetBin_CR_Esc add use of backspace key.
code	1	Minor	W	GetLong - Input comment in header should be where number is "stored".
code	7	Minor	W	Line 423 use all caps for TRUE.
code	4	Minor	W	Rework session report so that is fits on 1 page.
code	11	Minor	M	Line 664 - comment call of PrtOCL to indicate all output info is printed with single call to function. Also PrtLIC.
code	14	Minor	W	Spec change - need start line number for voucher input instead of Top or Bottom.
code	11	Minor	M	GetLIC - comment variable usage.
code	4	Minor	W	Replace lines 679 - 683 with clr_disp_code(LIC_line); display_code('prompt',EOL); Then allow GetLICFields to set cursor.
code	4	Minor	W	Lines 686/692 - Tabs should stay within a line. Remove the check for RT, Left Tab. Place it in GetLICFields. Also GetOCL.
code	5	Minor	W	Use single letter variables such as i, j for counters only. e.g. lines 705, 730.

Code Inspection for: Executive Options 6, 7, 8 Cont.

Inspection Type	Pr Type	Severity	Category	Description
code	1	Minor	W	SessionMenu header input should be session number selected.
code	1	Minor	W	Change name of clear_memory_2 to restore_defaults.
code	2	Minor	E	Line 961/999 remove the declaration of i, it is not used on the routine.
code	4	Minor	W	Prompt for Voucher T/B should be in config_input not GetTorB.
code	14	Minor	W	Spec error - if doc size 3.5 = Yes do all questions concerning voucher. Else skip to the purchaser label.
code	4	Minor	E	Delete line 1063 & 1082. Do not need key=NULL;
code	2	Minor	E	Delete xpos & key from the declarations of config_output. They are not used in the module.
code	14	Minor	W	Spec change - Use Check Defaults will default to N for output types.
code	4	Minor	E	Delete line 1138. It is redundant to ask if answer = yes or answer = no when the answer can only be yes or no. Config.com must be reworked to match specification.

Code Inspection for:Document Printing and Executive Options 4,9,10,13,14
Held 1/12/89
Total Prep Time 3 hrs/2 participants
Total Inspection Time 9 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category	Description
				convert.plm
code	1	Minor	W	Update all module headers so the calling statement matches the PL/M calling statement and parameter list.
code	1	Minor	W	Header for tens2str should indicate translation from 0 to 99 not 1 to 99.
code	1	Minor	W	Header for money2str should indicate the format of the number is implied decimal with no commas.
code	4	Minor	E	Delete lines 328/329 they are redundant.
				prndrv.plm
code	11	Minor	M	Comments for line spacing should say that it is the number of steps to advance or reverse for the LF or backup.
code	11	Minor	E	Move prn_head comments to included file.
code	11	Minor	M	Add comments for count34.
code	2	Minor	E	In the literals which mask field attributes, delete all but GRAPHICS because they are not used.
code	9	Minor	W	In prnt_xdata use ch instead of c for the character variable.
code	11	Minor	M	Line 295 comment offset = 16
code	1	Minor	M	Add the positioning is for both the check and voucher to header for goto_pos.
code	7	Minor	W	Line 312 use literal CR instead of 13.
code	7	Major	W	In general use literals when a value is used several times, when it could change or when the listing would be better understood with a literal.
code	9	Minor	E	Delete reset_fdatr and setup_fdattr, they are not used.
code	1	Minor	W	Change header for gprint to reflect what is actually happening.
code	4	Minor	E	Remove prt_code,prt_data & prt_crlf, they are redundant.
				prndoc.plm
code	1	Minor	M	Included explanation for printing check and check/voucher combinations in header.
code	11	Minor	M	Comment GAMT_LEN to indicate purpose and how to change in the future.
code	2	Minor	E	Delete MAXOCL it is already in the included file.

Code Inspection for:Document Printing and Exec Options 4,9,10,13,14 Cont.

Inspection Type	Pr Type	Severity	Category	Description
code	5	Major	W	In general put no list compiler directive in the included files not the file it is included in. This allows the listing of which included files are used but does not show the contents of the include file.
code	1	Minor	W	Header descr for extract_numstr should indicate that non-digit characters are filtered from amt_str not num_str.
code	2	Major	W	Line 651 xstart & ystart should be aux.
code	4	Minor	M	print_signature should check to see if suppression is necessary before calculation to see if greater than the suppress amt.
Haroon				Call set_printer_params in printer_init.
code	1	Minor	M	The header for print_ldollar should indicate large graphic amt.
code	4	Minor	W	The printer should be initialized and paper loaded before calling print_document.
code	4	Minor	W	If possible move sort_prn_fd so that the sorting is done once per session instead of for each doc.
code	11	Minor	M	For end statements on lines 1099/1100 comment them with the do or do case which is being ended.
code	4	Major	W	Void check needs to consider the format of the output. If there is no large graphic amount, void in the small graphic amount area. Use the location of the standard or custom word amount for the other voids.
code	4	Major	M	print_document should send back status. exec.plm
code	4	Minor	W	Line 1210 should be if answer = NO not if key = NO.
code	4	Minor	W	Delete line 1222 - 1224 and substitute a call to prepare_doc.
code	1	Minor	W	Option 9 header should be from 1 - MAXSESSION not 0 - MAXSESSION so pass-through printer is not included in session selection.
Haroon				Line 1446 should be PRESS RETURN OR ESC
code	4	Minor	W	Status should be index in 1538/1539.
code	4	Minor	W	A sample document field should contain the maximum number of characters which will be sent for that field.
code	4	Minor	E	Delete line 1569.
Haroon				Haroon - the display version must incorporate the release date.
code	4	Minor	W	config_com must be set up so MODEM cannot be selected.

Code Inspection for: Signature and Fixed Input/Output Types Loading
Held 2/6/89
Total Prep Time 2 hrs/2 participants
Total Inspection Time 3 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category	Description
				CONSOLE
code	1	Minor	M	Explain how and why the master side console routines are different than the standard CICID software.
code	4	Major	W	L83 ACK should be NAK.
				FIXLOAD
code	2	Minor	E	Fixin include file - Mac_payroll and acc should be deleted.
code	2	Minor	W	Fixin include file - PayeeL1IC should be Pay1LIC.
code	2	Minor	W	fixin include file - PayeeOCL1 and PayeeOCL2 should be Payee1OCL and Payee2OCL respectively.
code	14	Minor	W	When naming fixed input types, use complete company or package name and abbreviate acc payable or payroll.
code	2	Minor	W	fixout include file - Special message font should be 12 not 17 for Cashier Check and Money Order types.
code	3	Minor	W	The pointer for inlist should be inlist_ptr not tag_list.
code	1	Minor	M	select_fixin include in header that at least 1 fixed type must be in code memory.
code	11	Minor	M	L203 Comment.
code	2	Minor	W	Change variable reload to session_deactivated.
code	1	Minor	M	Indicate the style number and time it takes to print that signature in the function headers for signature loading styles.
code	1	Minor	M	Indicate how the single density loading is accomplished.
code	11	Minor	M	L429 Comment.
code	11	Minor	M	L453 Comment.

Code Inspection for: The Parser
Held 1/10/89
Total Prep Time 5 hrs/2 participants
Total Inspection Time 6 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category		Description
Code	1	Minor	M	p1	In parser header explain how to invoke the parser and where it is invoked.
Code	11	Minor	M	p2	Explain in declaration comments what constitutes an item and how the ItemIndexes and ItemLen are used.
Code	2	Minor	W	p2	MAXLINEBYTES should be lit 132 because CR and LF are not stored.
Code	2	Major	W	p2	The maximum number of items should be used instead of MAXLIC for the number of elements in the item indexes and length variables.
Code	11	Minor	W	p2	Comments for ItemsLen should refer to the length found not max length.
Code	11	Minor	W	p2	Comments for Item Indexes should not refer to item_ptrs because this variable is not based.
Code	9	Minor	E	p2	Remove variables Found, MustBeFound and Dum they are not needed.
Code	9	Minor	W	p2	Constant FORMFEED should be in the include file.
Code	7	Minor	E	p2	Do not use unnecessary globals.
Code	11	Minor	W	p2	Distinguish between input document and output document for clarity.
Code	11	Minor	M	p3	Shows that FINDITEMS works on the current line and 1 line at a time.
Code	4	Minor	W	p3	When determining the length of an item, use CHARS as the maximum length.
Code	4	Minor	E	p4	Delete module EXTRACTDATE. EXTRACTALPHANUMERIC can be used.
Code	1	Minor	M	p6	Include possible return codes in function header.
Code	4	Major	W	p6	Set up GetLine routine so that CNTRL key is acknowledged at document boundaries only. Either before or after a document is completely received and printed.
Code	5	Minor	W	p6	Use literals on line 440 instead of hardcoded numbers.
Code	4	Minor	W	p7	If an overflow is found when reading in a line, the check should be voided.
Code	4	Minor	W	pG	The variable attribute may not be necessary and should be removed.

Code Reinspection for: Areas of Major Change
Held 02/02/89
Total Prep Time 5 hrs/3 participants
Total Inspection Time 4.5 hrs/ 3 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category	Description
				EXEC.PLM
code	1	Minor	E	p3 InitOutTypeOCL header says there is an output type but there is not.
code	1	Minor	W	Gen All modules should have one header and they should be per standard.
code	4	Minor	W	p12 GetLong if value is 101 shows 1 1.
code	1	Minor	W	Gen Spacing guides should be in header not preceding module header.
code	11	Minor	M	p14 Line 859 the constant 12 should be commented.
code	7	Minor	W	Gen In final listing all procedure headers must be proceeded by a $EJECT.
code	4	Minor	W	p22 InMenuSession and OutMenuSession if key = ESC just return the key.
code	9	Minor	W	Gen Move Get_TorB to STD.LIB.
code	14	Minor	W	p31 Config_input, Prompt for VOUCHER STARTINGG has extra G.
code	14	Minor	M	Config_output, prompt for EMPL/VEND/BR is missing the work LABEL.
code	8	Major	M	p33 Display the last response entered for USE CHECK DEFAULTS. Suggest using a comparison of the values stored and the check defaults.
code	8	Minor	W	Make option 8 prompting for sessions like options 6 & 7.
code	14	Minor	E	config_session remove the period "." from the NO INPUT TYPES prompt.
code	4	Minor	W	p35 line 1602, if ret_code is ESC or = NO return instead of printing report.
code	4	Minor	W	p35 Do not init the printer if the report is not going to be printed.
code	8	Minor	W	p35 Use Prtch instead of prtcode for printing a single character in line 1621.
code	1	Minor	M	p39 config_com, comment that the modem is not currently supported.
code	2	Minor	W	p41 OKI and EPSON display information and inclusion as supported printers needs to be in an include file.
				POWERUP.PLM
code	2	Minor	M	Change procedure name DocRead to ParseDoc.
code	8	Minor	M	DOCSLEFT = ParseComplete
code	8	Minor	M	Gen Need to flush keyboard buffer if a CNTRL is received while online.

Code Reinspection for: Areas of Major Change

Inspection Type	Pr Type	Severity	Category	Description
code	4	Minor	M	Lines 519/520 & 579/580 Use clr_displ_code instead of separate clear and display code.
code	7	Minor	W	Version information is in the wrong format. STD.LIB
code	4	Minor	W	LongToAscii the number 101 will be displayed as 1 1.
code	7	Minor	W	p16 Use all caps for constants.
code	3	Minor	E	p17 CompressStr do not pass len.
code	4	Minor	W	p19 GetCompName does not return the compressed string.

Systems Test Inspection for: Power Up, Session Selection, Pass Through Printer, Exec Entry, Options 1,2,3,5, and 8.
Held 12/21/88
Total Prep Time 1.5 hrs/3 participants
Total Inspection Time 4 hrs/ 4 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category		Description
systst	6	Minor	M	p1	Add if lister is accidently turned on with T4400 hardware machine will hang up.
systst	2	Minor	W	2	In general info all sessions must be deactivated AND the pass through printer must be turned off.
systst	2	Minor	M	p3	Number 5, step 7, add that taking the pass through printer off line may cause 1 or more characters to be lost. Losing them does not affect the test validity.
systst	3	Major	M	p4	Add a test to print macola accounts payable and payroll check with pass through printer.
systst	3	Minor	M	p5	Add case of all spaces for exec code. It should be accepted.
systst	3	Minor	M	p5	Add case to check keys other than alpha-numeric keys. (i.e. function keys)
systst	2	Minor	W	p6	Remove "when available" description. Instruct tester to skip section.
systst	2	Minor	M	p6	Expand description in 2 step 3.
systst	6	Minor	M	p7	Add a pretest preparation for test 9. This should include clearing all memory.
systst	3	Minor	M	p7	Add verification for test 9.2.3
systst	6	Minor	M	p8	Add a verification step for 9.3.4.

Systems Test Inspection for: Part Two - Exec Options 4,6,7,9,10,11,12, 13,14 and Document Printing
Held 1/11/89
Total Prep Time 3 hrs/3 participants
Total Inspection Time 8 hrs/ 4 participants
Reinspection Required? N

Inspection Type	Pr Type	Severity	Category		Description
systst	3	Major	M	pG	Add test for speed comparison in printing with the packages and through the 4400.
systst	6	Minor	M	p1	Indicate that the pass through printing instructions are in Systems Test 1 Appendix A.
systst	6	Minor	M	p1	All memory should be cleared in pretest preparation section.
systst	2	Minor	W	p1	Only 1 grid is printed.
systst	3	Minor	M	p1	1.3 - Expand instructions to indicate paper must be loaded.
systst	3	Minor	E	p2	2.2.1/3.2.1 Delete this step it duplicates other tests.
systst	2	Minor	W	p3	2.2.3 - expand instructions for ESC out.
systst	4	Minor	W		Change spec so that when IT and COL are both >0 nothing is done.
systst	4	Minor	M		Change specification so the no session prompt matches the other no session prompt.
systst	3	Minor	M	p5	5.2 - add a third step which sends a large .lst file to the pass through printer session.
systst	3	Minor	M	p6	6.2 - add a 4th step which sends the minimum date.
systst	3	Minor	M	p6	Add a time check to validate approximation to real time.
systst	6	Minor	M	p8	Instruct the operator what to call the session.
systst	3	Minor	M	p8	Run a large quantity of intypes 9 and 10 with the macola package.

Specification Change Analysis
Performed 2/7/89

Pr Type	Severity	Category	Numb	Page	Description
				Release 1.4 02/03/89 pas	
4	Minor	M	1.	1	Non-real-time clock systems will be prompted with the time and date the machine was turned off at not the default time.
4	Minor	M	2.	13	Lister print when the printer buffer is cleared.
6	Minor	W	3.	15	Option 5, pass through printer will have a fixed session name of PASS THROUGH PRINTER. The option will provide the mechanism for turning it only off/on.
8	Minor	W	4.	16,18	If the NOT USED type is selected for input/output definition go directly to the rename with the name all blanks.
6	Minor	M	5.	19	Allow the user to determine the voucher size if the voucher is printed.
1	Minor	W	6.	22	Selection of the session for configuration will work like input/output selection.
6	Minor	W	7.	22	Move the activation prompt for session types to after the input/output type selection.
4	Minor	M	8.	23	If there are no input/output types defined prompt the user and upon RETURN go to the SELECT OPTION prompt.
2	Minor	W	9.	29	Line 6 with an N goes to line 9.
7	Minor	M	10.	29	Loading a signature of type 0 will blank the signature area. This will in effect print a blank signature.
2	Minor	M	11.	30	Prompt changes in predefined input/output type loading. Including an indication that if they were loaded the sessions will all be deactivated.
2	Minor	W	12.	GEN	To improve readability all selection, label and naming prompts have a dash "-" following the prompt.
7	Major	M	13.	17,21	Add PAYEE 2 per marketing request.

Specification Change Analysis Cont.

Pr Type	Severity	Category	Numb	Page	Description
				Release 1.3 01/18/89 pas	
8	Major	M	1.	4,2,15 23,24	Add ESC to load paper prompt allowing an ESC to get out of printing when there is no paper loaded.
3	Minor	W	2.	8	Only 1 format for journal listing of the document will be used.
3	Minor	W	3.	9	Journal listing for document will show current date and time.
7	Minor	W	4.	14	Clear all memory clears all RAM memory including signature and predefined input and output types.
8	Major	W	5.	16	In both user input and output type selection instead of scrolling through up to 10 empty slots only 1 empty slot will be shown at a time. All previously defined user types will be displayed with up/down arrows.
7	Minor	E	6.	17	Delete check for IT and CL both being greater than zero. A later version will be a total warning report.
1	Minor	W	7.	24	In printing sample document the prompt for no sessions will be the same as no sessions when going on line.
7	Major	M	8.	29,30	In signature pack when loading predefined input/output types scroll through the available types and allow them to choose up to 10.
6	Minor	W	9.	5	Changed RS232 READY/BUSY cabling to match what was implemented.
				Release 1.2 12/16/88 pas	
8	Minor	M	1.	2	Display all 20 characters of session type when in the online mode. Prompt will then be CNTRL FOR OFF LINE/Session IDccccccccc
6	Minor	W	2.	16	Replace voucher top or bottom for input session query to voucher starting line query. Prompt will be VOUCHER STARTING LINE XX.
2	Minor	W	3.	19	If there is no voucher included in the output type skip to line 9 instead of line 8.

Specification Change Analysis

Pr Type	Severity	Category	Numb	Page	Description
				Release 1.1 12/8/88 pas	
4	Minor	W	1.	1	Power-up: Upon power up all data not printed will be cleared.
5	Minor	W	2.	all	Do not use Y/N for selecting types (session, input output) use Up and Down arrows and RETURN to select.
4	Minor	M	3.	3	If no sessions are active, display prompt NO SESSIONS ACTIVE PRESS RETURN.
6	Major	M	4.	6	Add Ready/Busy protocol for RS232.
1	Minor	W	5.	all	All type names will be 20 characters.
8	Minor	M	6.	8	Pass through printer: S to STOP/S to START instead of ESC.
4	Minor	M	7.	9	Added lister printing upon entry to the EXEC mode if lister is configured.
6	Minor	W	8.	10	Executive required will default to Y.
6	Minor	W	9.	13	After clearing all memory, require the executive to turn the power off/on the reboot the system.
6	Minor	W	10.	15	Add RENAME input type prompt and Type new name to input and output types.
4	Minor	W	11.	15	If the number of lines in the voucher is 0, assume no voucher is transmitted.
8	Minor	W	12.	15	Ask only 1 date question. If MM/DD/YY is N assume MONTH XX,XXXX.
7	Major	W	13.	16	Add a column position to the input data location. Column position is used when the field starts in a specific column. Item is used then the field is proceeded by a constant number of fields that have various lengths.
8	Minor	W	14.	17	Use blank fill instead of place-holder characters.
8	Minor	W	15.	18	Ask only one doc length question. If 3.5 is N, assume 3.4.
9	Minor	W	16.	22	Default session type names to NOT USED 1 - NOT USED max.
8	Minor	W	17.	24	Rework communications selection to work like the 4500.
3	Minor	W	18.	25	Lister print the new time in option 11 if lister is configured.
7	Minor	M	19.	27	Include a new option to display the version information.
7	Minor	W	20.		Move the report to the session type. This allows all input and output types to be included in a report.

Error Analysis For: CICID Testing
March 16, 1989

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 02 (01/16/88)
Error	4	Maj	W	Activation and deactivation of the solenoid needs to occur at the proper times for printing documents and printing sample documents. (In T03 the delay time is not quite long enough. There is a slight tearing of the continuous forms holes.)
Error	2	Min	W	The font for printing the voucher needs to be a selectable option in the out type definition.
Error	3	Maj	M	When the print buffer is cleared it should be noted on the lister printer (if lister printer is configured).
Error	3	Maj	M	When all memory is cleared it should be noted on the lister printer (if lister printer is configured).
Error	3	Min	W	The time displayed upon power on without a real time clock should be the last machine time not the default (01/01/80).
Error	2	Min	W	Option 5 change the pass through printer session name to PASS THROUGH PRINTER and do not allow the user to change it.
Error	3	Maj	W	Option 10 the selection for Modem and RS232 should show that the modem is N and RS232 is Y. Also entry should not be allowed.
Error	3	Min	W	Option 10, the correct operation for a two item line with Y/N responses is to leave the cursor under the left-most and changes to that item automatically changes the right-most item.
Error	2	Maj	W	After loading the in and out fixed types from the signature pack all sessions should be deactivated.
Error	2	Min	M	If in or out fixed types are loaded in the signature pack the user should be warned that all active sessions will become inactive. Prompt as follows: LOADING COMPLETE - SESSIONS DEACTIVATED If no in or out fixed types were loaded the final prompt should be: LOADING COMPLETE
Error	5	Maj	W	The DOCS LEFT warning is being displayed when all documents/printer data has been printed.

Error Analysis For: CICID Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 02 (01/16/88) cont.
Error	4	Min	W	Print font for pass through printer should be independent from document printing. Assume normal print unless specifically changed to compressed. Once changed to compress all pass through printer sessions will be in compressed until the power is turned off. At power on assume normal printing for pass through.
Error	4	Min	W	When time is set at power-up a time of 12:33 PM shows up as 12:33 AM.
Error	1	Maj	M	Option 7 - Voucher Printed Prompt is not being displayed when a voucher has been selected.
Error	4	Maj	W	The amount in words for 1,000,000.00 was ONE MILLION THOUSAND DOLLARS.
Error	3	Min	W	Per spec - When the date and time are set they should be lister printed.
				Software Version Test 03 (02/03/89)
Error	4	Maj	W	Purchaser 2 is being placed in Purchaser 1 variable location.
Error	4	Min	M	If power off occurs while the session name is blank it is left blank. Editing of the session name should occur in a temporary variable.
Error	4	Min	W	Prtch routine should look for both an ACK and an error.
Error	1	Maj	E	Console is locking up on long lister printing requests.
Error	1	Min	E	An extra current date & time are being lister printed upon power-up.
Error	1	Min	W	When more than 19 characters are sent to the lister printer it must wrap-around in a human readable format. It is backwards.
Error	1	Min	W	Pass-through printer is being displayed for selection in print sample document, option 9. It should not be a valid selection.
Error	3	Min	W	In signature load, the default to loading the signature should be Y. It is N.
Error	4	Min	W	If paper is removed after the parser has been called, the docs left warning is not correctly displayed.

Error Analysis For: CICID Testing

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 04 (02/15/89)
Error	4	Min	W	A Document with 1 billion as the amount was correctly voided (we cannot print values this large). The lister type printed the amount however. Fix will print the words VOID surrounded by stars at the start of the lister tape for a voided document.
Error	4	Min	W	3.4 inch documents are creeping up approx. 1/4th in. over a 240 document pack.
Error	4	Min	W	The delay to recheck the paper sensor is not quite long enough. One machine correctly determines when paper has not been loaded. The other machine does not.
				Software Version Test 05 (02/27/89)
Error	2	Min	W	All paper out conditions which prompt NO PAPER/LOAD PAPER ... should reprompt if a RETURN is pressed when paper has not been loaded. An ESC should be the only way to exit.
Error	5	Maj	W	RealWorld and Macola terminate a session in ways that leave docs to print or (RealWorld only) print an extra void document.

Error Analysis For: Official Items Dispenser Release 3 Testing
March 17, 1989

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 2 (9/23)
Error	4	Maj	W	Time AM/PM is not working correctly. An A can be entered but it always displays/lister prints as PM.
Error	4	Maj	W	Purchaser name is not being printed.
Error	1	Min	W	When a duplicate operator code is entered and CLEAR is pressed, the first operator with that code is displayed, it should be the second (one which is duplicated).
Error	1	Maj	W	When memory has been filled the prompt request the DAILY or CLEAR key be pressed. If the CLEAR key is pressed the prompt is redisplayed (i.e. continual loop). We should return to the ENTER OPERATOR CODE prompt.
				Software Version Test 3 (9/30)
Error	6	Maj	W	Machine hung saying amount over the operator amount. Was able to scroll through the information as if in local echo.
Error	4	Maj	W	While Lister printing for a multiple document session, the purchaser was printed the following number of times: 461-465 6 times, 466 5 times 467 4 times, 468 3 times 469 2 times, 470 1 time All but 470 had other characters printed.
Error	4	Maj	W	Machine hung. Operator was prompted with amount over the document limit. Amount entered was not over the limit. Entered the executive code and machine hung.
Error	3	Min	W	Documents whose amounts are greater than the top of the fee table are given the top of the fee table in the reports. (Seen on CC expected on others.)
Error	3	Min	W	Previous reports should print the date requested not the current date. (Seen on Previous Daily, assumed on other previous reports.)
Error	4	Maj	W	The modem communications is not working after 15 of machine.
				Software Version Test 4 (10/8)
Error	1	Min	W	Use of new editor will necessitate updating all manuals.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 4 (10/8) Cont.
Error	4	Maj	W	Document with an address partially lister prints, does not print the document but is included in the daily reports.
Error	3	Maj	W	Documents cannot be voided. No match is always displayed.
Error	3	Maj	W	Pack logic does not work the following is valid in full pack logic: Pack size 1000 Start 890000 End 100000000999
Error	3	Min	W	Aborting a serial number entry leaves the new serial numbers instead of retaining the serial numbers which were displayed before changes.(Does not appear to be fixed in Test 5.)
Error	1	Maj	W	When communications is enabled and the document array is filled, there is no message displayed and the machine locks up.
Error	3	Min	M	Agent number (id) requires a 7 character field with the new editor it has only a 2 character field.
Error	3	Min	M	Supervisor report does not show documents voided by the executive under options 17 and 18.
Error	1	Min	W	When displaying an other check type as in the Discount etc., the scrogg marks should be replaced by blanks. (When defining the document id the scrogg marks should be there as placeholders.)
Error	1	Min	M	In the Supervisors report a blank line is needed between the OP ID line and the next operators totals.
				Software Version Test 5 (10/9)
Error	1	Min	E	For a non-signature document, if a purchaser is not entered do not print the address label (for the purchaser) on the check.
Error	4	Maj	W	A document with a signature and address hangs at the point where the purchaser should be printed. The lister printer correctly prints the purchaser and address.
Error	1	Min	M	When in the Overtype mode a dot should appear in the first position of the display.(Requires a new Console)

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 5 (10/9) Cont.
Error	1	Min	W	In option 1, enter up code - If an entry has been made in a field a CLEAR should clear the field and the second CLEAR return to the Select Option prompt. Currently the first CLEAR returns to the Select Option prompt.
Error	3	Maj	M	Type of document is missing on Journal tape. Those documents are not included in the Daily report.
				Software Version Test 6 (10/15)
Error	3	Maj	M	Type of document is missing on Journal type. Testing stopped.
				Software Version Test 7A (10/19) 2000 byte doc array.
Error	4	Maj	W	Wrap-around exact fill 1 day, hung at the daily report. Second time it hung after the daily report at the enter fee type prompt. Test stopped.
				Software Version Test 7 (10/19)
Error	4	Maj	W	Graphic amount is scooting approx. 1/5 in.
Error	4	Maj	W	The text amount for 100001.00 is coming up as ONE HUNDRED ONE DOLLARS AND 00 CENTS. Should be ONE HUNDRED THOUSAND ONE DOLLARS AND 00 CENTS. (Problem exists in Release 2).
Error	3	Min	W	Option 19 use of the up and down arrows is not according to spec. See spec for details
Error	4	Maj	W	The text amount for 16 million dollars does not have the word DOLLARS listed.
Error	1	Maj	W	When five incorrect operator codes are entered and the ENTER EXECUTIVE PASS CODE prompt is displayed, if a CLEAR is pressed the ENTER OPERATOR CODE prompt is displayed allowing the operator to try again. It should remain in the ENTER EXECUTIVE CODE prompt until a correct exec code has been entered.
Error	1	Min	M	Previous reports for which no documents are stored should lister print the report type, date requested, current time and bank id (same as Release 2) before printing the NO DOCUMENTS STORED FOR date.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 8 (10/30)
Error	4	Min	W	If the first time the ENTER DATE OF REPORT prompt is displayed an N is entered the 4500 does not respond. You can type in a date after the N and a report will be transmitted for that date.
Error	4	Maj	W	The first daily report transmitted under REPORT has garbage for the date says NO DAILY REPORT. The data is not transmitted.
Error	4	Min	W	If the password is less than 9 characters (eg. 5 characters - 12345) and the password sent from the external device matches the password for those characters but appends others to the end (eg. 12345abc), the password is accepted. It should not be accepted.
Error	4	Min	W	If an invalid option is entered at the ENTER OPTION prompt the OID will not accept any subsequent valid options.
Error	4	Min	W	The date entered at the ENTER DATE OF REPORT prompt must be 6 digits long to be accepted. The current software sometimes accepts less (eg. 1187), other times it does not.
Error	1	Min	W	COMMUNICATIONS IN PROGRESS prompt has been displayed twice when trying to enter the document dispensing mode. Monitor for this problem after changes in where this message is displayed are implemented.
				Software Version Test 9 (11/13/87)
Error	1	Min	W	In a multiple document session all documents should print before returning to the ENTER OP CODE prompt.
Error	1	Min	W	Change paper out message to: NO DOCS LOAD DOCS THEN PRESS CLEAR
Error	1	Min	W	12:59PM increments to 13:00PM it should go to 1:00AM. 12:59AM increments to 1:00AM it should be 1:00PM
Error	4	Maj	W	Delete oldest day did not set the pointers correctly.
Error	4	Maj	W	In a ten document session the 3wk. document partially printed, a printer control code %C405 was printed, the 4th doc was missing and the 5th doc had an enlarged signature. DOCS 6 - 10 printed correctly.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 9 (11/13/87) Cont.
Error	1	Min	W	If communications is enabled, you are outside of the auto-answer time, an operator code has been entered and the phone key is pressed communications comes up. The phone must also be ringing for communications to come up.
Error	1	Min	W	If the phone key has been pressed the SELECT DOCUMENT TYPE prompt is still displayed and a document can be defined while communications is up. You should go back to the ENTER OP CODE prompt after the PHONE key is pressed.
Error	1	Min	W	If communications is disabled, a valid operator code is entered, the phone key is pressed and then communications is enabled, upon exiting the executive mode communications comes up. i.e. The PHONE key is remembered even through document dispensing.
Error	1	Maj	W	When communications is up you can enter the executive mode. You should not be able to.
				Software Version Test 10 (11/17/87)
Error	4	Min	W	Document array was 2018 bytes instead of 40000 bytes.
Error	1	Maj	W	Other check type with no signature was scooting.
				Software Version Test 11 (11/20/87)
Error	4	Maj	W	Console:Lister Printing problems. Probable cause, length of time spent in main loop.
Error	4	Maj	M	Previous Daily, Supervisor and Shift reports for dispensed documents are not found. Lister prints header and NO DOCUMENTS STORED FOR date.
Error	1	Min	W	The CLEAR key should reset the operator pass code to nulls. This is the only way to get back to less than 8 characters in the code. (The nulls following the entered pass code should also be retained when less than 8 characters.
Error	4	Maj	W	Several Wrap-around problems have been discovered.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 12 (12/11/87)
Error	1	Min	W	Authorization set for DOC limit only. For Money Orders, if the amount is > the operator max, the AMT D.E. OP LIMIT ENTER PASS CODE prompt is displayed. The operator limit should not be considered for this authorization type. Other document types work correctly.
Error	1	Min	W	For all authorization types, the Money Order message when the document amount is greater than the Money Order Max should be: AMT GT DOC LIMIT PRESS CLEAR TO REENTER.
Error	1	Min	W	For Dual Authorization, when the combined total of the dispensing operator and the authorizing operator maximum amounts is less than the document amount the message displayed should be: AMT GT DUAL LIMIT ENTER PASS CODE.
Error	1	Min	W	For Exec Only authorization, amounts over the op limit should prompt AMT GT OP LIMIT ENTER EXEC CODE.
Error	4	Min	M	Other check type id's of less than 15 characters have FF's instead of blanks to fill out the field.
Error	4	Maj	W	Dates longer than 6 digits for which the first 6 digits form a valid date are treated as a valid date. They should be rejected and the ENTER DATE OF REPORT prompt redisplayed.
Error	3	Maj	W	Dates with more than one close out are not being found.
Error	4	Min	W	10 non-nak characters received by the 4500 when going into xmodem should abort the session. (The same action as 10 naks while waiting for an ack after a block transmission.)
Error	4	Min	W	Communications may be brought when the phone has rung within the auto answer period while the 4500 is in a non-accessible area. Upon returning to the ENTER OP CODE prompt the 4500 bring communications up and then time aborts when the link is not established. It effect remembering the ring after the phone has stopped ringing.
Error	4	Min	W	While transmitting a 100 document day, documents after number 48 were trashed. (Could have been a problem putting the 4500 under emulation.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version Test 13
Error	3	Min	W	The lock message sent by the T4500 to the polling computer is not as specified.
Error	4	Maj	E	Trash has been found in a null authorizing operator field.
				Software Version Test 14
Error	4	Maj	W	With no paper in the document printer tried to print a document. After loading documents and pressing CLEAR, the machine has unpredictable behavior. It appears that the stack has been trashed.
Error	1	Maj	W	The check for out of paper should not occur after the last document of a session. If it is the end of a pack only the executive can load a new pack and the software does not allow access to the executive at the load documents prompt.
Error	4	Maj	M	During a transmission of a wrap around test some of the documents were omitted. The reports/previous reports were correct.
				Software Version Test 15
Error	4	Min	W	If the auto answer window is changed to a time which would be outside of the window (in reference to the current machine time), an attempt to initiate communications prior to the 1 minute timer interrupt will be successful. It should not allow communications to come up.
Error	4	Min	W	After changing the password, initiated communications. After the first ring the lister printed COMM UP LOST CARRIER 5 x in a row. Could have been a modem problem. Could not repeat.
Error	1	Maj	W	Initiation of Communications was attempted while in the Executive mode and did not come up (correct). Subsequently, when trying to define a document the display read COMMUNICATIONS IN PROGRESS as if the ring had set the comm up.
Error	1	Min	W	Problems with the PHONE key.
				Software Version Test 16
Error	1	Maj	W	All months fail to roll over to the next month. e.g. 01 31 becomes 01 32.
Error	1	Maj	W	If the PHONE key is pressed erroneously, and the operator answers Y at the ANSWER PHONE Y/N prompt. The system is hung. It never times out.
Error	4	Min	W	Delays should be increased for VERSION, LOCK , UNLOCK, and NO UNTRANSMITTED DAYS.
Error	4	Maj	W	RS232 does not come up.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version 16.02 (02/02/88)
Error	4	Maj	W	System locked up at power up.
Error	4	Maj	W	Real-Time Clock problems.
				Software Version 16.03 (02/11/88)
Error	1	Min	W	The executive code is displayed as it is entered, it should not be.
Error	1	Maj	W	Machine locks up when a date is not entered for a previous report.
Error	1	Min	W	Minor discrepancies reporting, - Previous Shift report when a day is not found is formatted differently than a date which is found, and PRINTING REPORT report does not come up when a supervisors report is printing.
				Software Version Test 17 (02/22/88)
Error	4	Maj	W	The third document of a multiple document session printed only a few control characters and started skipping lines. Power was turned off and the next multiple document session was dispensed correctly. (This is a graphics problem seen only on one machine.)
Error	4	Maj	W	Machine locked up when a daily report was taken. Found document array partially trashed. (Seen only once in testing.)
Error	6	Maj	W	After a long report had completely printed, system hung at the ENTER OP CODE prompt. Problem probably an interprocessor communications hang-up.
Error	4	Maj	W	When a day closed out with now documents was reclaimed the software was in a continual loop.
Error	6	Maj	W	An F1 typed at the polling computer in response to the ENTER OPTION prompt reboots the T4500.
Error	4	Maj	W	If a date requested at the ENTER DATE OF REPORT prompt is valid but not in memory, a type 0 - header record is not being sent.
Error	1	Min	W	If a 4500 is unlocked and and UNLOCK command is given, the polling device receives the MACHINE UNLOCKED message but the message is not lister printed.
				Software Version Test 18 (03/02/88)
Error	4	Maj	W	Got an over-sized signature for part of a document in the middle of a multiple document session.
Error	6	Maj	W	4 times the display was scrambled.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version R03.00-001.001-
Error	1	Maj	W	Delete Oldest day was not printing totals.
Error	4	Maj	W	In Executive Option 28:Communications Configuration at the Start and Stop Time entry, a time of 8:xx PM or 9:xx PM was not stored correctly.
Error	1	Min	W	In Executive Option 28:Communications Configuration at the Password entry, clearing a password set it to 9 spaces instead of 9 nulls (place-holders). After clearing the password you could never enter a password less than 9 characters.
Error	1	Min	M	When attempting to void a document which has been included in a daily closeout the DOCUMENT CANNOT BE VOIDED PRESS CLEAR prompt is displayed. R03.01-001.001- has an extra character in this line losing the "R" in CLEAR.
Error	Error	Maj	W	The RS232 polled communications would not work with a purchased null modem cable on some personal computers.
				Software Version R03.02-001.001-
Error	4	Maj	W	When power was removed prior to completion of printing a document, the document record was not retained in memory. The serial number had been used and no record of the document being dispensed was maintained.
				Software Version R03.03-001.001-
Error	1	Maj	W	When operator 32 was defined with an operator ID less than 10 characters, the operator maximum amount for operator 1 was altered such that any document dispensed by operator 1 for less than the document maximum could be dispensed without authorization. The same problem could lose one character of another operator ID when operator ID's have less than 10 characters entered.
Error	4	Maj	W	If no purchaser was entered (available on other check types only) and one was not required, extra characters are transferred into the document array. This would not generally pose a problem but could cause a problem if this document was the last document before a report or the last document stored in memory before reclaiming the oldest day for more document storage.

Error Analysis For: Official Items Dispenser Release 3 Testing Cont.

Inspection Type	Pr Type	Severity	Category	Description
				Software Version R03.04-001.001-
Error	4	Maj	W	In polled communications, EOL (FF) was being transmitted in the XMODEM block. This also resulted in the oldest day in the machine being appended to the date requested.
Error	4	Min	W	In polled communications, the routine which prepares the document type defn record (type "1") did not check to see if the other check types were set to "Y" when deciding to send the other check id's. The problem arose when the other check types were set to "Y" but did not have an id defined.
Error	4	Maj	W	Printer hang-ups have been reported from the field. The problem arose when an interrupt occurred between the time the call to send the character to the printer was made and the time the need for acknowledge flag was set. When an interrupt occurred in this time frame the printer had already received the character and reset the flag, but the flag had not been set. When the flag was subsequently set, the printer was waiting for a character and the software was waiting for the flag to be reset. At that point the system was hung.
Error	4	Min	W	Upon reviewing the autoanswer code a typographical error could have caused the autoanswer time to be incorrectly false. This problem was not reported from the field.

APPENDIX C

SPECIFICATION INSPECTION

MODULE Date	CI CID Sept 19, 88	CI CID Reinspection Oct 5, 88	Specification Error Totals
System Description			
Major-M			0
Major-W	2	1	3
Major-E			0
Major-Total	2	1	3
Minor-M	3	9	12
Minor-W	4	4	8
Minor-E	1		1
Minor-Total	8	13	21
TOTAL	0	14	24
Data Requirements			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W		1	1
Minor-E			0
Minor-Total	0	1	1
TOTAL	0	1	1
Environ/Operation Constraints			
Major-M			0
Major-W	1		1
Major-E			0
Major-Total	1	0	1
Minor-M		3	3
Minor-W	2		2
Minor-E			0
Minor-Total	2	3	5
TOTAL	3	3	6
System Security			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0

MODULE	CI CID	CI CID Reinspection	Specification Error Totals
Date	Sept 19, 88	Oct 5, 88	
Est. of Next Stages			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
Ref-Related Material			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
User Interface			
Major-M	1	1	2
Major-W	1		1
Major-E		1	1
Major-Total	2	2	4
Minor-M	2	8	10
Minor-W	4	6	10
Minor-E	5	4	9
Minor-Total	11	18	29
TOTAL	13	20	33
Other			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
Total All Errors	26	38	64
Reinspection Required	Y	N	
Preparation Time	24.0	19.5	43.5
Inspection Time	30.0	24.5	54.5

INITIAL DESIGN INSPECTION

Module	Initial Design Of CI CID
Date	Oct 10, 1988
System Flow Diagram	
Major-M	
Major-W	0
Major-E	
Major-Total	0
Minor-M	
Minor-W	2
Minor-E	
Minor-Total	2
TOTAL	2
Data Dictionary	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	1
Minor-W	
Minor-E	
Minor-Total	1
TOTAL	1
Data Flow Diag or STD	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	14
Minor-W	1
Minor-E	
Minor-Total	15
TOTAL	15
Module Interface Link	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	
Minor-E	
Minor-Total	0
TOTAL	0

Module	Initial Design Of CI CID
Date	Oct 10, 1988
Module Descr	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	1
Minor-W	
Minor-E	
Minor-Total	1
TOTAL	1
Performance	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	
Minor-E	
Minor-Total	0
TOTAL	0
Spec Clarification	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	
Minor-E	
Minor-Total	0
TOTAL	0
Spec Incorrect	
Major-M	
Major-W	1
Major-E	
Major-Total	1
Minor-M	
Minor-W	
Minor-E	
Minor-Total	0
TOTAL	1

Module	Initial Design Of CI CID
Date	Oct 10, 1988
Standards	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	
Minor-E	
Minor-Total	0
TOTAL	0
Other	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	
Minor-E	
Minor-Total	0
TOTAL	0
Total Errors	20
Reinspection Required	N
Preparation Time	2.0
Inspection Time	6.0

DETAILED DESIGN INSPECTION

Module	Power-up & Pass-through Printer	Exec Entry Exec Options 1,3,11	Reinspection Exec Entry Opt 1,3,11
Date	Oct 13, 1988	Oct 17, 1988	Oct 19, 1988
Data Flow Diagram			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	2		
Minor-W			
Minor-E			
Minor-Total	2	0	0
TOTAL	2	0	0
Data Dictionary			
Major-M	1	0	0
Major-W			
Major-E			
Major-Total	1	0	0
Minor-M	2		
Minor-W	1	2	1
Minor-E	1		
Minor-Total	4	2	1
TOTAL	5	2	1
Logic			
Major-M	3	3	2
Major-W	2	2	
Major-E			
Major-Total	5	5	2
Minor-M	3		
Minor-W	1	1	2
Minor-E		1	1
Minor-Total	4	2	3
TOTAL	9	7	5
Data Area Usage			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0

Module	Power-up & Pass-through Printer	Exec Entry Exec Options 1,3,11	Reinspection Exec Entry Opt 1,3,11
Date	Oct 13, 1988	Oct 17, 1988	Oct 19, 1988
Test and Branch			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Return Codes/Messages			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Register Usage			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
External Linkages			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0

Module	Power-up & Pass-through Printer	Exec Entry Exec Options 1,3,11	Reinspection Exec Entry Opt 1,3,11
Date	Oct 13, 1988	Oct 17, 1988	Oct 19, 1988
More Detail			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W	2		
Minor-E			
Minor-Total	2	0	0
TOTAL	2	0	0
Standards			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Header/Comments			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M		1	
Minor-W	1		
Minor-E	1		
Minor-Total	2	1	0
TOTAL	2	1	0
Initial Design Doc			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0

Module	Power-up & Pass-through Printer	Exec Entry Exec Options 1,3,11	Reinspection Exec Entry Opt 1,3,11
Date	Oct 13, 1988	Oct 17, 1988	Oct 19, 1988
Specification			
Major-M	0	0	0
Major-W		2	
Major-E			
Major-Total	0	2	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	2	0
Maintainability			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Performance			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Other			
Major-M	0	0	0
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W	1		
Minor-E			
Minor-Total	1	0	0
TOTAL	1	0	0
Total Errors	21	12	6
Reinspection Required	N	Y	N
Total Preparation Time	2.0	1.5	1.0
Total Inspection Time	6.0	1.5	4.5

Module	I/O - Check and Voucher Data Structures	Detailed Design Error Totals
Data Flow Diagram		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		2
Minor-W		0
Minor-E		0
Minor-Total	0	2
TOTAL	0	2
Data Dictionary		
Major-M	0	1
Major-W		0
Major-E		0
Major-Total	0	1
Minor-M	5	7
Minor-W		4
Minor-E	1	2
Minor-Total	6	13
TOTAL	6	14
Logic		
Major-M	0	8
Major-W		4
Major-E		0
Major-Total	0	12
Minor-M		3
Minor-W		4
Minor-E		2
Minor-Total	0	9
TOTAL	0	21
Data Area Usage		
Major-M	0	0
Major-W	1	1
Major-E		0
Major-Total	1	1
Minor-M		0
Minor-W	1	1
Minor-E		0
Minor-Total	1	1
TOTAL	2	2

Module	I/O - Check and Voucher Data Structures	Detailed Design Error Totals
Test and Branch		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
Return Codes/Messages		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
Register Usage		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
External Linkages		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0

Module	I/O - Check and Voucher Data Structures	Detailed Design Error Totals
More Detail		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		2
Minor-E		0
Minor-Total	0	2
TOTAL	0	2
Standards		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
Header/Comments		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M	3	4
Minor-W		1
Minor-E		1
Minor-Total	3	6
TOTAL	3	6
Initial Design Doc		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0

	I/O - Check and Voucher Data Structures	Detailed Design Error Totals
Specification		
Major-M	0	0
Major-W		2
Major-E		0
Major-Total	0	2
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	2
Maintainability		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
Performance		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
Other		
Major-M	0	0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		1
Minor-E		0
Minor-Total	0	1
TOTAL	0	1
Total Errors	11	50
Reinspection Required	N	
Total Preparation Time	1.0	5.5
Total Inspection Time	4.5	16.5

Module	Mainline & Pass-through Printer	Reinspection Mainline & Pass-thru Prnt	Console Intrf Exec Entry, Exec Opt 1,2,3
Date	Nov 8, 1988	Nov 29, 1988	Nov 9, 1988
Function			
Header			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W		1	
Minor-E			
Minor-Total	0	1	0
TOTAL	0	1	0
Declares and Defines			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			1
Minor-W	1	1	4
Minor-E	4		2
Minor-Total	5	1	7
TOTAL	5	1	7
Entry and Exit Links			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			1
Minor-E			
Minor-Total	0	0	1
TOTAL	0	0	1
Logic			
Major-M	3		3
Major-W	1		2
Major-E	1		
Major-Total	5	0	5
Minor-M			4
Minor-W	1	3	2
Minor-E			1
Minor-Total	1	3	7
TOTAL	6	3	12

Module	Mainline & Pass-through Printer	Reinspection Mainline & Pass-thru Prnt	Console Intrf Exec Entry, Exec Opt 1,2,3
Date	Nov 8, 1988	Nov 29, 1988	Nov 9, 1988
Language Usage			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W		2	1
Minor-E			
Minor-Total	0	2	1
TOTAL	0	2	1
Memory Usage			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Standards			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	1		
Minor-W	1		
Minor-E			
Minor-Total	2	0	0
TOTAL	2	0	0
Performance			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			1
Minor-Total	0	0	1
TOTAL	0	0	1

Module	Mainline & Pass-through Printer	Reinspection Mainline & Pass-thru Prnt	Console Intrf Exec Entry, Exec Opt 1,2,3
Date	Nov 8, 1988	Nov 29, 1988	Nov 9, 1988
Maintainability			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W	4		
Minor-E			1
Minor-Total	4	0	1
TOTAL	4	0	1
Detail Design Error			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Comments			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			5
Minor-W	1		
Minor-E			
Minor-Total	1	0	5
TOTAL	1	0	5
Other			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	1	1	
Minor-W	1	1	1
Minor-E	2		
Minor-Total	4	2	1
TOTAL	4	2	1
Total Errors	22	9	29
Reinspection Required	Y	N	Y
Total Prep Time	3.0	2	3.0
Total Inspection Time	7.5	3.0	9.0

Module	Reinspection Console Intrf Exec Entry etc	Reinspect 2 Console Intrf Exec Entry etc	Exec Options 6,7,8
Date	Nov 23, 1988	Nov 30, 1988	Dec 13, 1988
Function Header			
Major-M			1
Major-W			
Major-E			
Major-Total	0	0	1
Minor-M			
Minor-W			4
Minor-E			
Minor-Total	0	0	4
TOTAL	0	0	5
Declares and Defines			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W	1	1	1
Minor-E	1	1	2
Minor-Total	2	2	3
TOTAL	2	2	3
Entry and Exit Links			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Logic			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	2		1
Minor-W	3	1	5
Minor-E			2
Minor-Total	5	1	8
TOTAL	5	1	8

Module	Reinspection Console Intrf Exec Entry etc	Reinspect 2 Console Intrf Exec Entry etc	Exec Options 6,7,8
Date	Nov 23, 1988	Nov 30, 1988	Dec 13, 1988
Language Usage			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			1
Minor-E			
Minor-Total	0	0	1
TOTAL	0	0	1
Memory Usage			
Major-M			
Major-W			1
Major-E			
Major-Total	0	0	1
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	1
Standards			
Major-M			
Major-W			1
Major-E			
Major-Total	0	0	1
Minor-M			
Minor-W			1
Minor-E			
Minor-Total	0	0	1
TOTAL	0	0	2
Performance			
Major-M	1		
Major-W			
Major-E			
Major-Total	1	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	1	0	0

	...nspection Console Intrf Exec Entry etc	Reinspect 2 Console Intrf Exec Entry etc	Exec Options 6,7,8
Date	Nov 23, 1988	Nov 30, 1988	Dec 13, 1988
Maintainability			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W	1		1
Minor-E			
Minor-Total	1	0	1
TOTAL	1	0	1
Detail Design Error			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Comments			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	3	2	3
Minor-W			1
Minor-E			
Minor-Total	3	2	4
TOTAL	3	2	4
Other			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			3
Minor-E			
Minor-Total	0	0	3
TOTAL	0	0	3
Total Errors	12	5	28
Reinspection Required	Y	N	N
Total Prep Time	3.0	3.0	4.5
Total Inspection Time	4.5	1.5	6.0

Module	Document Parser	Document Printing	Areas of Major Change and Problem Fixes
Date	Jan 10, 1989	Jan 12, 1989	Feb 2, 1989
Function Header			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	1	3	1
Minor-W		6	2
Minor-E			1
Minor-Total	1	9	4
TOTAL	1	9	4
Declares and Defines			
Major-M			
Major-W	1	1	
Major-E			
Major-Total	1	1	0
Minor-M			1
Minor-W	1		1
Minor-E		2	
Minor-Total	1	2	2
TOTAL	2	3	2
Entry and Exit Linkages			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			1
Minor-Total	0	0	1
TOTAL	0	0	1
Logic			
Major-M		1	
Major-W	1	1	
Major-E			
Major-Total	1	2	0
Minor-M	1	1	1
Minor-W	3	7	6
Minor-E	1	3	
Minor-Total	5	11	7
TOTAL	6	13	7

Module	Document Parser	Document Printing	Areas of Major Change and Problem Fixes
Date	Jan 10, 1989	Jan 12, 1989	Feb 2, 1989
Language Usage			
Major-M			
Major-W		1	
Major-E			
Major-Total	0	1	0
Minor-M			
Minor-W	1		
Minor-E			
Minor-Total	1	0	0
TOTAL	1	1	0
Memory Usage			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Standards			
Major-M			
Major-W		1	
Major-E			
Major-Total	0	1	0
Minor-M			
Minor-W		1	3
Minor-E	1		
Minor-Total	1	1	3
TOTAL	1	2	3
Performance			
Major-M			1
Major-W			
Major-E			
Major-Total	0	0	1
Minor-M			2
Minor-W			2
Minor-E			
Minor-Total	0	0	4
TOTAL	0	0	5

	Document Parser	Document Printing	Areas of Major Change and Problem Fixes
Date	Jan 10, 1989	Jan 12, 1989	Feb 2, 1989
Maintainability			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W	1	1	1
Minor-E	1	1	
Minor-Total	2	2	1
TOTAL	2	2	1
Detail Design Error			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			
Minor-W			
Minor-E			
Minor-Total	0	0	0
TOTAL	0	0	0
Comments			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M	2	5	1
Minor-W	3	1	
Minor-E			
Minor-Total	5	6	1
TOTAL	5	6	1
Other			
Major-M			
Major-W			
Major-E			
Major-Total	0	0	0
Minor-M			1
Minor-W			1
Minor-E			1
Minor-Total	0	0	3
TOTAL	0	0	3
Total Errors	18	36	27
Reinspection Required	N	N	N
Total Prep Time	5.0	3.0	5.0
Total Inspection Time	6.0	9.0	4.5

Module	Signature & Fixed Type Loading	Code Inspection Error Totals
Date	Feb 6, 1989	
Function Header		
Major-M		1
Major-W		0
Major-E		0
Major-Total	0	1
Minor-M	4	9
Minor-W		13
Minor-E		1
Minor-Total	4	23
TOTAL	4	24
Declares and Defines		
Major-M		0
Major-W		2
Major-E		0
Major-Total	0	2
Minor-M		2
Minor-W	4	15
Minor-E	1	13
Minor-Total	5	30
TOTAL	5	32
Entry and Exit Links		
Major-M		0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W	1	2
Minor-E		1
Minor-Total	1	3
TOTAL	1	3
Logic		
Major-M		7
Major-W	1	6
Major-E		1
Major-Total	1	14
Minor-M		10
Minor-W		30
Minor-E		7
Minor-Total	0	47
TOTAL	1	61

Module	Signature & Fixed Type Loading	Code Inspection Error Totals
Date	Feb 6, 1989	
Language Usage		
Major-M		0
Major-W		1
Major-E		0
Major-Total	0	1
Minor-M		0
Minor-W		5
Minor-E		0
Minor-Total	0	5
TOTAL	0	6
Memory Usage		
Major-M		0
Major-W		1
Major-E		0
Major-Total	0	1
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	1
Standards		
Major-M		0
Major-W		2
Major-E		0
Major-Total	0	2
Minor-M		1
Minor-W		6
Minor-E		1
Minor-Total	0	8
TOTAL	0	10
Performance		
Major-M		2
Major-W		0
Major-E		0
Major-Total	0	2
Minor-M		2
Minor-W		2
Minor-E		1
Minor-Total	0	5
TOTAL	0	7

Module	Signature & Fixed Type Loading	Code Inspection Error Totals
Date	Feb 6, 1989	
Maintainability		
Major-M		0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		9
Minor-E		3
Minor-Total	0	12
TOTAL	0	12
Detail Design Error		
Major-M		0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		0
Minor-W		0
Minor-E		0
Minor-Total	0	0
TOTAL	0	0
Comments		
Major-M		0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M	3	24
Minor-W		6
Minor-E		0
Minor-Total	3	30
TOTAL	3	30
Other		
Major-M		0
Major-W		0
Major-E		0
Major-Total	0	0
Minor-M		3
Minor-W	1	9
Minor-E		2
Minor-Total	1	14
TOTAL	1	14
Total Errors	16	201
Reinspection Required	N	
Total Prep Time	2.0	33.5
Total Inspection Time	3.0	54.0

SYSTEMS TEST INSPECTION

Module	Systems Test 1 Mainline, Pass-Through Printer Exec 1,2,3,5,8	Systems Test 2 Document Printing Exec 4,9,10,13,14	Systems Test Inspection Error Totals
Date	Dec 12, 1988	Jan 11, 1989	
Approach			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
Test Descr			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M	1		1
Minor-W	3	2	5
Minor-E			0
Minor-Total	4	2	6
TOTAL	4	2	6
Test Procedure			
Major-M	1	1	2
Major-W			0
Major-E			0
Major-Total	1	1	2
Minor-M	3	5	8
Minor-W			0
Minor-E		1	1
Minor-Total	3	6	9
TOTAL	4	7	11
Systems Spec			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M		1	1
Minor-W		1	1
Minor-E			0
Minor-Total	0	2	2
TOTAL	0	2	2

Module	Systems Test 1 Mainline, Pass-Through Printer Exec 1,2,3,5,8	Systems Test 2 Document Printing Exec 4,9,10,13,14	Systems Test Inspection Error Totals
Date	Dec 12, 1988	Jan 11, 1989	
Hardware/Build Req			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
Operator Instructions			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M	3	3	6
Minor-W			0
Minor-E			0
Minor-Total	3	3	6
TOTAL	3	3	6
Messages			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
Other			
Major-M			0
Major-W			0
Major-E			0
Major-Total	0	0	0
Minor-M			0
Minor-W			0
Minor-E			0
Minor-Total	0	0	0
TOTAL	0	0	0
Total Errors	11	14	25
Reinspection Required	N	N	
Total Prep Time	1.5	3.0	4.5
Total Inspect Time	4.0	8.0	12.0

Inconsistent with Other Areas	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	3
Minor-E	
Minor-Total	3
TOTAL	3
Prompts/Instructions for Moving from Prompt	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	1
Minor-W	3
Minor-E	
Minor-Total	4
TOTAL	4
Lister Printing	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	3
Minor-E	
Minor-Total	3
TOTAL	3
Omissions	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	6
Minor-W	2
Minor-E	
Minor-Total	8
TOTAL	8

Deletions	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	2
Minor-E	
Minor-Total	2
TOTAL	2
Implementation Required Changes	
Major-M	1
Major-W	1
Major-E	
Major-Total	2
Minor-M	1
Minor-W	6
Minor-E	
Minor-Total	7
TOTAL	9
Marketing/Function Required Changes	
Major-M	2
Major-W	1
Major-E	
Major-Total	3
Minor-M	3
Minor-W	2
Minor-E	1
Minor-Total	6
TOTAL	9
Improved User Interface	
Major-M	1
Major-W	1
Major-E	
Major-Total	2
Minor-M	2
Minor-W	5
Minor-E	
Minor-Total	7
TOTAL	9
Superseded by Later Change	
Major-M	
Major-W	
Major-E	
Major-Total	0
Minor-M	
Minor-W	1
Minor-E	
Minor-Total	1
TOTAL	1
Total Changes to Specification	48

SYSTEMS TESTING ERROR ANALYSIS

Module Date	CI CID Systems Testing Jan 16 - March 10	OID Rel 3 Systems Testing Sept 23 - April 13
User Interface		
Major-M	1	
Major-W		14
Major-E	1	
Major-Total	2	14
Minor-M		3
Minor-W	2	20
Minor-E	1	1
Minor-Total	3	24
TOTAL	5	38
Spec Not Correct		
Major-M		
Major-W	1	
Major-E		
Major-Total	1	0
Minor-M	1	
Minor-W	3	
Minor-E		
Minor-Total	4	0
TOTAL	5	0
Function Not As Specified		
Major-M	2	2
Major-W	1	3
Major-E		
Major-Total	3	5
Minor-M		2
Minor-W	4	5
Minor-E		
Minor-Total	4	7
TOTAL	7	12
Problem in Implementation		
Major-M		2
Major-W	3	31
Major-E		1
Major-Total	3	34
Minor-M	1	1
Minor-W	7	13
Minor-E		
Minor-Total	8	14
TOTAL	11	48

Module Date	CI CID Systems Testing Jan 16 - March 10	OID Rel 3 Systems Testing Sept 23 - April 13
Unexpected Interaction W/External S/W Package		
Major-M		
Major-W	2	1
Major-E		
Major-Total	2	1
Minor-M		
Minor-W		
Minor-E		
Minor-Total	0	0
TOTAL	2	1
Inter-processor Communication Problems		
Major-M		
Major-W		4
Major-E		
Major-Total	0	4
Minor-M		
Minor-W		
Minor-E		
Minor-Total	0	0
TOTAL	0	4
Total Testing Errors	30	103

APPENDIX D

SPECIFICATION INSPECTION DETAIL REPORT

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. System Description_____							
2. Data Requirements_____							
3. Environmental/Operational Constraints_____							
4. System Security_____							
5. Estimate of Next Stages_____							
6. Ref to Related Material_____							
7. User Interface_____							
8. Other_____							
TOTAL							

REINSPECTION REQUIRED? __ (Y or N)
TOTAL PREPARATION TIME:
LENGTH OF INSPECTION:
NUMBER OF PARTICIPANTS:

INITIAL DESIGN INSPECTION MODULE DETAIL REPORT

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. System Flow Diagram							
2. Data Dictionary							
3. Data Flow and/or State Transition Diagrams							
4. Module Interface Linkages							
5. Module Description							
6. Performance							
7. Spec Clarification							
8. Specification Incorrect							
9. Standards							
10. Other							
TOTAL							

REINSPECTION REQUIRED? __ (Y or N)
TOTAL PREPARATION TIME:
LENGTH OF INSPECTION:
NUMBER OF PARTICIPANTS:

DETAILED DESIGN INSPECTION MODULE DETAIL REPORT

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. Data Flow Diagram							
2. Data Dictionary							
3. Logic							
4. Data Area Usage							
5. Test and Branch							
6. Return Codes / Messages							
7. Register Usage (ASM only)							
8. External Linkages							
9. More Detail							
10. Standards							
11. Header or Comments							
12. Init Design Document							
13. Specification							
14. Maintainability							
15. Performance							
16. Other							
TOTAL							

REINSPECTION REQUIRED? __ (Y or N)
TOTAL PREPARATION TIME:
LENGTH OF INSPECTION:
NUMBER OF PARTICIPANTS:

CODE INSPECTION DETAIL REPORT

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. Function Header							
2. Declarations and Defines							
3. Entry and Exit Linkages							
4. Logic							
5. Program Language Usage							
6. Memory Usage							
7. Standards							
8. Performance							
9. Maintainability							
10. Detail Design Error							
11. Comments							
12. Other							
TOTAL							

REINSPECTION REQUIRED? __ (Y or N)
TOTAL PREPARATION TIME:
LENGTH OF INSPECTION:
NUMBER OF PARTICIPANTS:

SYSTEM TEST INSPECTION MODULE DETAIL REPORT

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. Approach							
2. Test Description							
3. Test Procedure							
4. System Specification							
5. Hardware/Build Reqs							
6. Operator Instructions							
7. Messages							
8. Other							
TOTAL							

REINSPECTION REQUIRED? __ (Y or N)
TOTAL PREPARATION TIME:
LENGTH OF INSPECTION:
NUMBER OF PARTICIPANTS:

SPECIFICATION CHANGE ANALYSIS REPORT FORM

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. Inconsistent W/Other Areas							
2. Prompts/Instruct Moving from Prompt to Prompt							
3. Lister Printing							
4. Omissions							
5. Deletions							
6. Implement Reqd Changes							
7. Marketing Req'd Changes							
8. Improved User Interface							
9. Superseded by Later Change							
TOTAL							

TESTING ERROR ANALYSIS REPORT FORM

DATE_____

MODULE:_____ COMPONENT/APPLICATION_____

PROBLEM TYPE:	MAJOR			MINOR			TOTAL
	M	W	E	M	W	E	
1. User Interface							
2. Specification Not Correct							
3. Function not as Specified							
4. Problem in Implementation							
5. Unexpected Interaction w/ External S/W Package							
TOTAL							