CHAPTER TWO
LITERATURE REVIEW

The literature review focuses on inspection methods and inspection process. This chapter emphasizes on the concepts and ideas on an effective inspection process. The discussion is subsequently followed by a comparison on the development approach, concepts and techniques of the existing tools in software code inspection.

2.1 INSPECTION PROCESS

Inspection process can be categorized into seven processes -- Fagan Inspection, Structured Walkthroughs, Humprey's Inspection, Gilb and Graham Inspection, N-Fold Inspection, Phased Inspection and Asynchronous Inspection (Knight & Meyers, 1993).

2.1.1 Fagan Inspection

Michael E. Fagan defined the original inspection process in 1976 (Fagan, 1976), with an update published ten years later (Fagan, 1986). A Fagan inspection team consists of four to six people, with each person having a well-defined role in the inspection. These roles include *moderator*, *author*, *reader* and *recorder*.

The *moderator* is the person in charge of the overall inspection. The responsibility of the moderator is to ensure that the inspection procedures are performed throughout the entire inspection process. This includes ensuring that the other inspection team members perform their roles to the best of their ability. The inspection requires the presence of the *author* of the product. The author can give invaluable help to the inspectors by answering questions pertaining to the intent of the document or work product. Generally, their main duty is to look for defects in the document or work product. However, at the inspection meeting, two inspectors are given special roles as a *reader* or a r*ecorder*. The *reader* is responsible for leading the inspection team through the inspection meeting by reading aloud small logical units, paraphrasing where appropriate. The *recorder's* task is to note all defects found, along with their classification and severity. Although Fagan indicates that the *moderator* accomplishes this task but another member of the team is usually chosen because of the workload involved can be quite high (Fagan, 1986).

Fagan also described five stages in the inspection process, depicted in Figure 2.1 (Fagan, 1976). The inspection begins with an *overview*, involving the entire team in the overview phase. The *author* describes the general area of work, and then gives a detailed presentation on the document produced. This is followed by distribution of the document itself, and any necessary related work to all team members. Then, each team member carries out an individual *preparation*, consisting of studying the document to gain an understanding of it. In this phase, each of the team members will inspect the work individually, looking for defects in the work product.

The next stage is the *inspection meeting*, involving all team members to discuss the possible defects in the work product. The *reader* paraphrases the document, covering all areas. During this process, *inspectors* can stop the *reader* and raise any issues they have covered, either in preparation or at the meeting itself. Then, the team discusses the issues until agreement is reached. If an issue is agreed to be a defect, it is classified as *missing*, *wrong* or *extra*. Its severity is classified as *major* or *minor*. At this point, the meeting moves on. No attempt is made to find a solution to the defect; this is carried out later. After the meeting, the moderator writes a report detailing the inspection and all defects found. Then, this report is passed to the author for *rework*, where the author carries out modifications to correct the defects found in the document.

After that in the *follow-up* phase, the moderator ensures that all the required alterations have been made. Then, the moderator decides whether the document should be partially or fully re-inspected and presented a report.

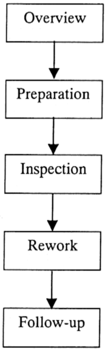

Figure 2.1: The original inspection process defined by Michael Fagan

The Fagan inspection process is summarized in Table 2.1. For each phase, the table shows the timing, the participants, the documents made available and the documents produced. Timing is either synchronous (S) or asynchronous (A).

Table 2.1: Summary of Fagan's inspection phases

Phase	Timing	Participants	Documents Used	Documents Produced
Overview	S	Moderator Author Inspector	Product Sources	
Preparation	A	Moderator Author Inspector	Product Sources Checklist	Individual defect logs
Inspection	S	Moderator Reader Recorder	Individual defect logs Product Sources Checklist	Master defect log Inspection report
Rework	-	Author	Product Sources Master defect log	
Follow-up	-	Moderator	Product Master defect log	Follow-up report

2.1.2 Structured Walkthroughs

Another popular method is Yourdon's Structured Walkthrough, in which inspection tends to be less formal and less rigorous (Yourdon, 1986). Yourdon defines seven possible participant roles including *coordinator, scribe, presenter, reviewers, maintenance oracle, standards bearer* and *user representative.*

The *coordinator's* planning and organizing the walkthrough and also takes the role of moderator during the walkthrough meeting. The role of the *scribe* is to take notes on the walkthrough, including any defects found and suggestions made. The *presenter* is tasked with introducing the product and is usually the author of the product. The role of *presenter* is optional. There are also a number of *reviewers,* whose task is to find defects in the product. The remaining three roles are *maintenance oracle* (who is concerned with future maintenance of the project), the *standards bearer* (who is concerned with the adherence to standards) and the *user representative* (whose task is to ensure that the product meets the user's needs). Although Yourdon describes these as separate roles, it can be seen that they are simply reviewers with special responsibility.

The Structured Walkthrough process is shown in Figure 2.2. The first phase is *organization* that begins with the producer requesting a walkthrough. The *producer* supplies the appropriate documentation to the *coordinator*, who then distributes it to all participants. The *coordinator* also arranges a time and place for the *walkthrough* and contacts all participants to confirm the arrangement (Waldstein, 1976). The participants then spend time preparing the walkthrough by reviewing the product.

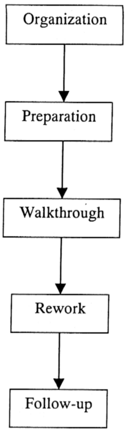

Figure 2.2: The Structured Walkthrough process presented by Yourdon

During this stage, the *producer* should be available to answer questions and help participants familiarize themselves with the documents. The *walkthrough* itself begins with the *presenter* providing an *overview* of the product. The length of this *overview* will depend on the familiarity of the participants with the documents (Yourdon, 1986; Waldstein, 1976).

After that, the product is presented entirely and the *reviewers* have the opportunity to make any comments about the product. Comments from the *preparation* phase that required no explanation can be passed straight to the *producer* and the *scribe*. As *reviewers* present other comments, the *producer* may ask for clarification, but should not spend time arguing about the validity of the comments. As with other review methods, there should be no discussion on how each defect may be corrected.

The *walkthrough* phase should last between thirty to sixty minutes and finishes with a vote on the status of the product (Yourdon, 1986). After the *walkthrough*, the *coordinator* prepares a management summary and a list of detailed comments. These comments are distributed to all participants. The *producer* makes the required alterations to the product during the *rework* phase, deciding on the validity of each comment and seeking guidance from the other participants as appropriate (Yourdon, 1986). Finally, a *follow-up* phase is conducted to ensure that the required changes have been made to the product.

The phases, participants and documents present during a structured walkthrough are summarized in Table 2.2. Timing is either synchronous (S) or asynchronous (A). The documents used and produced during each phase are also listed.

Table 2.2: Summary of the Structured Walkthrough phases

Phase	Timing	Participants	Documents Used	Documents Produced
Organization	-	Coordinator Producer		
Preparation	A	Coordinator Producer Reviewer	Product	Individual list
Walkthrough	S	Coordinator Producer Reviewer	Product Individual list	Master defect log Inspection report
Rework	-	Coordinator Producer	Product Master defect log	Summary
Follow-up	-	Coordinator	Product	

2.1.3 Humphrey's Inspection Process

The inspection process described by Humphrey (Humprey, 1989) is very similar to that described by Fagan. However, there are also some major differences. The inspection team consists of three persons with the roles of *moderator*, *producer* and *reviewer*. The roles described are virtually identical in name to those described by Fagan, but the actual process is different.

The process is depicted in Figure 2.3. The *planning* stage allows for the selection of participants and preparation of entry criteria. The *overview* stage is identical to that of Fagan's. It is during the *preparation* stage that the first deviation from Fagan's method where defect detection is deferred until the meeting. These defect logs are then passed to the *producer* for what could be termed the *analysis* phase, where the individual logs are analyzed and consolidated into a single defect list (Humprey, 1989).

At the *inspection* meeting itself, the *producer* addresses each defect and asks the *reviewer* to clarify the meaning of each defect. A list of agreed defects is produced. The meeting is followed by the typical post inspection activities of *rework* and *follow-up*.

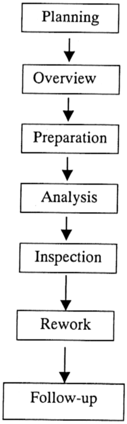

Figure 2.3: The inspection process described by Watts Humphrey

The inspection process is summarized in Table 2.3, which describes the possible timing of each phase and the participants involved. Timing is either synchronous (S) or asynchronous (A). The documents used and produced during each phase are also listed.

Table 2.3: Summary of Humprey inspection phases

Phase	Timing	Participants	Documents Used	Documents Produced
Planning	-	Moderator Producer		Inspection objectives Participant list
Overview	S	Moderator Producer Reviewer		
Preparation	A	Reviewer	Product Checklist Standards	Defect logs Preparation report
Analysis	-	Producer	Defect logs	Consolidated log
Inspection	S	Moderator Producer Reviewer	Consolidated log	Master defect log Inspection report Inspection summary
Rework	-	Producer	Product Master defect log	
Follow-up	-	Moderator Producer	Product Master defect log	

2.1.4 Gilb and Graham Inspection

One of the most comprehensive texts on software inspection is that of Gilb and Graham (Knight & Meyers, 1993). The method they describe is obviously based on Fagan's work. However, it also incorporates other's lessons. One such lesson is the defect prevention process described by Jones (Jones, 1985).

There are three defined roles in this type of inspection which are the *leader, author* and *checker*. The *leader* is in charge of the overall process and is responsible for the planning and conducting the inspection. The *author* is the person who produced the product document. Besides attending the logging meeting, the *author* should also take part in checking. The remaining team members are *checkers,* whose duty is simply to find and report defects in the document. During the logging meeting, one of the *checkers* is assigned the role of scribe and logs the issues found during the inspection (Kelly et al, 1992).

The process is illustrated in Figure 2.4. It begins by ensuring that some *entry* criteria have been satisfied. These will ensure that the inspection is not wasted on a document that is fundamentally flawed. This is followed by inspection *planning,* where the *leader* determines inspection participants and schedules the meeting. This phase produces a master plan for the entire inspection. After that, in the *kick off* phase, the relevant documents are distributed and the *leader* briefed about master plan. Then, participants are assigned to roles and the goals are set. The goals include checking rates to be met and expected defect rates.

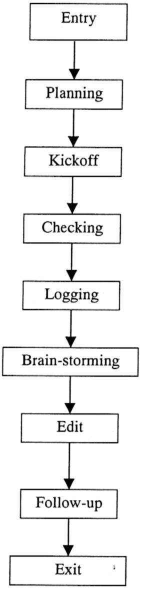

Figure 2.4: The inspection process described by Gilb and Graham

Checking comes in the next phase where each *checker* works alone to discover defects in the documents. The potential defects are recorded for presentation in the logging phase where logging meetings are scheduled. The logging session is a highly structured meeting where potential defects found by the *checkers* are collected. The emphasis here is to log in as many issues as possible. The inspection leader moderates the meeting and ensures that the discussion is kept focused and criticisms are minimized.

The meeting is followed by a *brainstorming* session to record process improvement suggestions. After all potential defects have been logged, the *author* takes the issue list and performs an edit on the product. At this point, all related issues are classified as defects. During the *follow-up* phase occurs where the *leader* ensures that the edit phase has been properly executed. Finally, some *exit* criteria must be satisfied before the inspection can be declared complete. These criteria typically consist of such item as checking rates, which must be within certain limits and predicted number of defects left in the document (Gilb & Graham, 1993).

The fundamental difference between this process and that of Fagan's is that defect detection was carried out before the group meets. The meeting consists of raising and logging defects. The process is similar to Humphrey's, but with partial similarity, since the producer is not expected to analyze the defect logs before the meeting. Instead, each *checker* simply presents the defects they derive from the documents. Process improvement is also not an explicit feature of Humphrey's process neither at the *entry* nor at the *exit* phase. A summary of Gilb and Graham's process is given in Table 2.4.

Table 2.4: Summary of Gilb and Graham inspection phases

Phase	Timing	Participants	Documents Used	Documents Produced
Entry	-	Leader	Entry criteria	
Planning	-	Leader		Master plan
Kickoff	S	Leader		Goal
		Author		
		Checker		
Checking	A	Leader	Product	Issue lists
		Author	Sources	
		Checker	Standards	
			Checklist	
			Procedures	
			Master Plan	
Logging	S	Leader	Product	Issue log
		Author	Sources	
		Checker	Standards	
			Checklist	
	-		Procedures	
			Master plan	
			Issue lists	
Brainstorming	S	Leader		Process improvements
		Author		
		Checker		
Edit	-	Author	Product	
			Issue log	
Follow-up	-	Leader	Product	
Exit	-	Leader	Exit Criteria	

2.1.5 N-Fold Inspection

The N-Fold inspection process is based on the notation that the effectiveness of an inspection can be improved by replicating it (Martin & Tsai, 1990). While two teams may individually find 40% to 50% of the total number of defects in the document, one team may find defects not found at all by the other and vice versa. There will be some overlap between the defects found but these newly founded defects could outweigh these advantages. By increasing the number of inspection teams, the overall percentage of defects found will gradually increase, until a point where the cost of finding more defects (i.e. using more teams) is greater than the benefits gained from removing those extra defects. This technique was originally designed to be used for user requirements documents, since the defects that were injected here are the most expensive to fix, but it could be used at any time. Removal of defects is of paramount importance in this process.

In addition, the personnel are required to hold each inspection. N-Fold inspection requires the services of a *coordinator* (moderator), whose task is to coordinate the teams and collect the inspection data. This is achieved by meeting with the moderator from each team. The description of N-Fold inspection is rather vague, apart from the essential feature that multiple inspections are carried out by independent teams. Therefore, the description is an extrapolated process, which would be necessary to effectively implement an N-Fold inspection (Martin & Tsai, 1990).

The process is shown in Figure 2.5. It begins with usual *planning* stage of a traditional inspection. However, this stage also includes deciding the number of teams to participate and other relevant roles to an N-Fold inspection. There will also be an overview stage to familiarize the participants with the context and contents of the document and the goals of the inspection. This is followed by a number of inspection stages, each of which is entirely independent. This means they use completely independent teams and possibly completely different inspection processes.

Using different *inspection* process improves the independence of each inspection and will hopefully find more defects in the document. Once each inspection has been completed, the process enters a *collation* phase, where the results of each inspection are tallied and collated by the *coordinator*. This stage produces a master list of defects that are given to the author for the traditional *rework* phase. This would be followed by a *follow-up* phase ensuring the required item has been addressed.

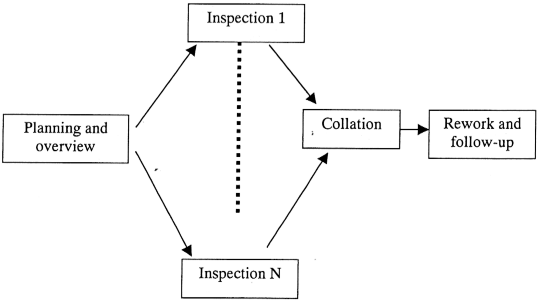

Figure 2.5: The N-Fold inspection process

The N-Fold inspection is summarized in Table 2.5. Note that many of the details of this process depend on the inspection method being used. Generally, the inspection produces defect lists, which must be collated into a single list. The only group phase is the collation stage, which can be asynchronous (A) or synchronous (S).

Table 2.5: Summary of N-Fold inspection phases

Phase	Timing	Participants	Documents Used	Documents Produced
Planning	-	Coordinator		
Overview	S	Coordinator Moderator Inspector	Product	Individual defect logs
Preparation	A	Moderator Author Inspector	Product Sources Checklist	
Inspection	-	-	Product	Defect list Master defect list
Collation	-	Coordinator Moderator	Defect list	
Rework	-	Author	Product Master defect list	
Follow-up	-	Coordinator	Product Master defect list	

2.1.6 Phased Inspection

The phased inspection technique was developed by Knight and Myers with the goal of permitting the inspection process to be rigorous, efficient in its use of resources and heavily computer supported (Myers, 1978). A phased inspection consists of an ordered set of phases, each of which is designed to ensure that the product possesses either a single, specific property or a small set of related properties. The phases are ordered so that each phase can be built on the assumption that the product contains properties which have been inspected in the previous phases. The properties that can be checked for are not necessarily those concerned purely with defects of functionality. They can include qualities such as reusability, portability and compliance with coding standards. The process, consisting of a number of phases, is depicted in Figure 2.6.

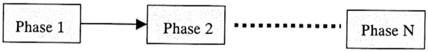

Figure 2.6: The Phased inspection process

There are two types of phase in phase inspection: *single-inspector* and *multiple inspectors*. A *single inspector* phase uses exact checklist, with the *inspector* deciding whether the product does or does not comply with each item. The phase cannot be completed until the product satisfies all checks. In contrast, *multiple inspector* phases are designed for properties, which cannot easily be described by the binary questions in single inspector checklists. The appropriate documents are initially distributed to each participant, who begins by examining this information and generating questions to clarify and improve the documentation. Then, each inspector inspects the product individually. This individual checking makes use of a checklist that is both application specific and domain specific though the question is not binary, as they are in the *single inspector* phase. The individual checking is followed by a meeting known as a *reconciliation* in which the inspectors compare their findings (Myers, 1978). Note that although it is not designed to do so, the *reconciliation* provides a further opportunity for defect detection.

Phased inspection is designed to allow experts to concentrate on finding defects that they have specialized knowledge of, thus, making more efficient use of human resources. For example, it may be more efficient to have *domain analyst* inspecting code for reusability, since the experts will have expert knowledge in that particular field. The phased inspection technique is summarized in Table 2.6, including the documents required and produced at each stage.

Table 2.6: Summary of the Phased inspection phases

Phase	Timing	Documents Used	Documents Produced
Single inspector	-	Product	Defect list
		Checklist	Completed checklist
Multiple inspector	A	Product	
examination		Sources	Question list
Inspection			
	A	Product	
		Sources	
Reconciliation		Checklist	Completed checklist
	S	Product	
		Completed checklist	

2.1.7 Asynchronous Inspection

Asynchronous inspection focuses on on-line meeting but this meeting is expensive as it is difficult to manage and conduct. However, one person must ensure that all team members are available at the same time and at the same place. This needs to arrange suitable meeting space. An alternative to an inspection meeting is to hold the entire inspection asynchronously, by providing some means of supporting discussion without the entire team being present at the same place and time (Johnson & Thahjono, 1993).

A similar system can be used for inspection. By allowing users to access an on-line version of the document, they can add comments to the document (indicating potential defects) using some types of annotation technology. These comments can be made available to other *inspectors*, who can further comment on the comments from the user. This procedure can continue until a consensus is reached on the status of the original comment, as either a defect or a non-issue. Once discussions have been completed on all comments, the inspection is complete and the document can enter *rework* phase. Such an inspection method has been implemented using a review tool called Collaborative Software Review System (CSRS). The tool implements an inspection method known as Formal Technical Asynchronous reviews method (FTArm) (Johnson & Thahjono, 1993).

As with traditional inspection, FTArm defines several roles: *moderator*, *producer* and *reviewer* (Johnson & Thahjono, 1993). The process itself is shown in Figure 2.7 and consists of seven phases. The first phase is *setup*, which involves choosing the members of the inspection team and preparing the document for inspection through CSRS. This involves organizing the document into a hypertext structure and entering it into the database. *Orientation* is equivalent to *overview* in the Fagan process and involves a presentation by the *author*. *Private review* is also similar to *preparation* in the Fagan's process.

A document is stored in a database as a series of nodes. For source code, these nodes would consist of functions, variable and other programming constructs. The reviewers read each source node in turn and have the ability to create new nodes containing annotations. These annotations can include issues indicating defects, comments, pertaining to the intention of the document, which may be answered by the producer and actions that indicate a possible solution to remove a defect (Johnson & Thahjono, 1993). When each reviewer has covered each node, the inspection moves on to the next phase. In *public review*, all nodes become public and inspectors can asynchronously examine each node and vote on its status. Votes cast confirm the issue, disconfirm the issue or indicate neutrality (Johnson & Thahjono, 1993).

Additional nodes can be created at this stage, immediately becoming publicly available. When all nodes have been resolved or if the moderator decides that further voting and on-line discussion will not be fruitful, the *public* phase is declared complete. During *consolidation*, the moderator analyses the results of the *private* and *public review* phases and summarizes unresolved issues. Then, the *moderator* decides whether a *group review meeting* is to be held to resolve the remaining issues. After that, the final inspection report produced by the *moderator* in the *conclusion* phase together with a metrics report.

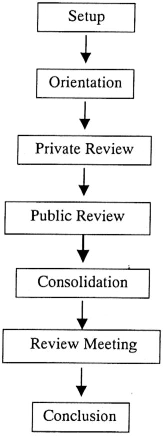

Figure 2.7: The FTArm Asynchronous inspection process

From this description, it can be seen that the FTArm is inherently computer-based. Computer support is essential for providing an asynchronous discussion environment. A summary of the FTArm process is given in Table 2.7. The timing of each phase can be synchronous (S) or asynchronous (A).

Table 2.7: Summary of the FTArm asynchronous inspection phases

Phase	Timing	Participants	Documents Used	Documents Produced
Setup	S	Moderator Producer		
Orientation	S	Moderator Producer Reviewer		
Private review	A	Moderator Producer Review	Product Checklist	Issues Comments Action
Public review	A	Moderator Producer Review	Product Issues Comments Actions	Issues Comments Actions
Consolidation	-	Moderator	Issues Comments	Consolidated issues Consolidation report
Review meeting	S	Moderator Producer Review	Product Issues Actions	
Conclusion	-	Moderator		Report

2.1.8 Summary

Initially, there appeared to be a large variety of inspection processes. The most radical variations come in the form of the N-Fold and Asynchronous processes. The N-Fold process is designed to increase the confidence in the quality of the document by using multiple and independent inspections team. Asynchronous inspection is designed to replace the need for a synchronous group meeting. In addition, asynchronous inspection is a good technique for inspection in a client server-computing environment. Most process difference occurs in terminology: the terms used for process phases, participants and documents all vary between methods, which can cause confusion when comparing processes. Finally, from the previous section; every inspection has three major stages: *Organization* (deciding on participants, timing and other details), *Detection* (performing the actual inspection and finding errors) and *Completion* (fixing the errors and checking the work done). These three broad stages are presented in all the inspection methods described earlier. The first and last stages only vary slightly between various inspection methods, while the major variations appear during the detection stages.

2.2 CODE INSPECTION TOOLS

Inspection is a cost-effective technique for finding and removing defects from documents produced during the software life cycle.

2.2.1 Existing Code Inspection Tools

This section describes the currently available tools to support code inspection. Although all tools described have the aim of improving the inspection process, each has its own approach.

2.2.1.1 Intelligent Code Inspection in a C Environment (ICICLE)

ICICLE is designed to support the inspection of C code (Brothers & Sembugamoorthy, 1992). This inspection tool is unique in making use of knowledge to assist in finding common defects. However, since the knowledge is of a very specific kind, ICICLE is less suitable for supporting inspection of other document types.

The tool is designed to support two phases of inspection: comment preparation (individual) and the inspection meeting itself. During the inspection meeting, the tool provides the functionality available in individual checking, supplemented by support for cooperative working (Brothers & Sembugamoorthy, 1990).

2.2.1.2 Distributed Code Inspection (DCI)

The Distributed Code Inspection (DCI) prototype proposed by Doherty and Sahibuddin is designed to implement the distributed inspection process (Doherty & Sahibuddin, 1997). The process starts with planning activity, where a synchronous or asynchronous method is chosen. In the synchronous model, the next activity is a kick-off meeting where the participants are briefed on the source code and the objectives of the inspection. In an asynchronous method, a briefing document is distributed by e-mail. Both models continue with individual preparation, where participants attempt to find defects in the code. Then, a group activity follows in both methods. Confusingly, the authors state that all participants must be available at the same time in both methods – surely this contradicts the asynchronous model. Both methods then end with a follow-up phase (Doherty & Sahibuddin, 1997).

2.2.1.3 Collaborative Software Review System (CSRS)

Collaborative Software Review System (CSRS) is probably the most flexible of all tools described here as it can be customized to support different variants of the inspection process (Johnson & Thahjono, 1993). This is accomplished using a process modeling language. This language has several facilities, including constructs for defining phases of the method; a construct for defining the role of each participant and constructs to define the artifacts used during the inspection (Johnson & Thahjono, 1993). The latter also includes support for checklists.

The language can also be used to define the user interface, as well as to control the type of data analysis carried out by CSRS. The description of CSRC presented here is based on its use to support asynchronous method of inspection known as Formal Technical Asynchronous review method (FTArm). This method consist of a phase of individual review of the product (where all comments are kept private), followed by public review (where all comments become publicly available and are discussed asynchronously) (Johnson & Thahjono, 1993).

2.2.1.4 hyperCode

HyperCode is a web-based tool and also designed to support different time, different place inspection. Preparation and collection are performed concurrently, while resolution of issues is performed during the rework phase (Harjumaa & Tervonen, 1998).

When an inspection has been started, the relevant inspectors are notified by e-mail. During a designated time span, an inspector uses a standard WWW browser to study an annotated code under inspection. All annotation is made public. When this period has elapsed, e-mail is again sent to the inspectors and the author may then examine annotations made and decide on the rework to be performed. On completion of the rework, the moderator is informed to verify the rework (Harjumaa & Tervonen, 1998).

2.2.2 Comparison of Existing Code Inspection Tools

The comparisons of tools are based on the functions and features of the code inspection tools. Table 2.8 gives the comparison of the features of code inspection tool and the function of each tool. The code inspection tools are Intelligent Code Inspection in a C or C++ Language Environment (ICICLE), Distributed Code Inspection (DCI), Collaborative Software Review System (CSRS) and hyperCode.

Table 2.8: Summary of the function and features of existing tool code inspection

Functions and Features of tools	ICICLE	DCI	CSRS	hyperCode
Functions	• Designed to support the inspection of C and C++ codes using UNIX	• Designed to implement their distributed inspection process.	• Can be customized to support different variants of the inspection process	• A WWW based tool also designed to support different time, different place inspection
Linked Annotation	*		*	*
Defect Classification	*		*	
Cross Referencing	*			
Automated Analysis	*			
Checklist				
Supporting Material	*	*		
Distributed Meeting		*		
Decision Support			*	
Data Collection	*		*	

2.2.3 Summary

Inspection is a cost-effective technique for finding and removing defects from documents produced during the software lifecycle. It is difficult to implement, very labour intensive and is a prime target for computer support. Therefore, existing code inspection tools should incorporate the following features:

a. Annotation Support

There should be a means for creating annotations which are linked to the part of document to which they refer. These annotations should be able to be given a type, indicating the purpose of that annotation or the type of fault which is describes, thereby giving more detailed metrics on the types of defects found.

b. Automatic Defect Detection

The tools should be able to detect simple types of defects to allow the inspectors to concentrate on finding more subtle defects. The tool should automatically produce annotations for the inspectors to review.

c. Checklist

Checklist should be available on-line for the inspector to use while inspecting the documents. Standards should also be available in a similar manner.

d. Distributed Meeting Support

There should be support for meetings between geographically disparate team members to allow a more expeditious inspection.

e. Metrics Collection

Collection of defect data should be done automatically by the tool to alleviate the problems of erroneous or incomplete data that can occur in manual data collection.

2.3 RESEARCH FRAMEWORK

In short, this research has significantly motivated the need for building a generic code inspection application as a way to ease the coding process. The details about this project are explained in chapters four, five and six. The study of the various applications listed above helped to identify the key characteristics of the proposed application tool. The various ways of providing support are highlighted in the last chapter.

The key characteristics that can be derived from the literature review are:

a. Implementing the application in web-based

   The main reason is that an Internet-based tool would have the ability to easily reach a large number of people, but on the other hand, using local network properties for efficiency reasons should also be considered whenever possible.

b. User-friendly interface

   This is important to let users feel that they are in control. GUIs have become the established alternative to traditional form of user interfaces.

c. Fagan Method

   This method allows the inspector to plan and inspect the code. The process is simple and easy to implement code inspection.