Malaysian Journal of Computer Science, Vol. 13 No. 1, June 2000, pp. 17-26

AN ASYNCHRONOUS SOFTWARE INSPECTION MODEL

Abdusalam F. Ahmed Nwesri and Rodina Ahmad
Faculty of Computer Science & Information Technology
University of Malaya
50603 Kuala Lumpur
Malaysia
Tel: +603 769-6368
Fax: +603 757-9249
email: nwesri@acm.org
rodina@fsktm.um.edu.my

ABSTRACT

This paper introduces an inspection model that discharges face-to-face meetings among inspectors in the software development group. The model aims at the elimination of the synchronous inspection meeting and the reduction of the asynchronous meeting by filtering out minor defects through the use of incontrovertible voting before the asynchronous meeting. The model goes through eight phases: initiation, overview, private checking, public checking, correlation, explicit voting, asynchronous meeting and rework and follow-up. The model is supported by a web-based tool named WASIT.

Keywords: ***Software inspection, Asynchronous inspection model, software inspection tools, software quality***

1.0 INTRODUCTION

Software inspection is a fundamental component of the software quality assurance process. It is a process whereby a group of software competent people critically checks a piece of software milestone for detecting defects. In contrast to testing, which is usually carried out at the final stage, the inspection process works as early as the first document in the software life cycle is ready. The process can be applied to user requirements, document software requirements, specification document, design specification or program codes documents. This process has been first introduced by Michael Fagan in the 1970's [1] and has been extensively used ever since. Inspection improves the quality of software products, such as understandability, portability, maintainability, testability, etc. Its success has always been demonstrated in many published articles.

A full inspection usually consists of three main activities including preparation, defect detection and collection, and defect correction. Since introduced, the detection and collection activity (or the inspection meeting) has been considered to be the essential element of the inspection process. Many other inspection models have been introduced, however, they maintain the inspection meeting as the main activity.

Further empirical studies in this decade have proven that the inspection meeting is of doubtful value [2]. Some of these studies have also revealed that inspection can be carried out, with the same results, without the inspection meeting. Consequently, some advanced models have been introduced.

Few asynchronous inspection models have been introduced in order to practice the inspection process without the inspection meeting. Those models have also looked forward to distribute the inspection process, allowing the inspection to be carried out from different places. The discussion activity (or the asynchronous meeting), as any other asynchronous activity, in these models however, takes more time than the detection and collection activity in the conventional inspection process. In contrast to the ordinary inspection meeting, which usually takes not more than two hours [1], the asynchronous meeting might last for days [3]. The number of defects passed to the asynchronous meeting proportionally increases the asynchronous meeting time. By minimizing that number, we believe that the time of such meeting can be reduced.

This paper aims to introduce an asynchronous inspection model whereby an additional explicit voting is added to resolve minor defects before the asynchronous inspection meeting takes place, thus minimizing the inspection period. We begin by presenting the original process. We follow this presentation by highlighting the contradictory points against the inspection meeting. We then describe the existing asynchronous models and finally present our model along with our argument.

2.0 EARLY WORK

The original inspection process supports five distinctive roles namely moderator, author, inspector, reader and the recorder. The moderator is the person in overall charge of the process. He/she is responsible for organizing and managing the inspection process; he/she elects and invites people to join the inspection, distributes inspection materials and moderates the inspection meeting. The author is the producer of the document under inspection. Usually, his/her task is to give any clarifications required

by the inspection team during the process. He/she also makes the changes necessary to the document. The inspector's main task is to detect any defects in the document under inspection. Any team member, except the author, can perform the roles of the reader and the recorder. The reader is supposed to paraphrase the document during the inspection meeting while the recorder records defects detected along with their classifications and severity.

The original process goes through five phases: *overview*, *preparation*, *inspection*, *rework* and *follow-up*. However in the updated version [4] the *planning* phase was added. The process is depicted in Fig. 1.

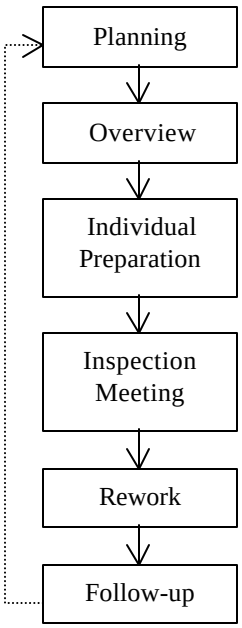

Fig. 1: Fagan's Inspection Process

In the *planning* phase, the entry criteria for the inspection materials is checked, the availability of the right participants and the time and place are arranged. In the following phase, *overview*, the author presents his product to the whole team. Then the document and any related work such as the source document and checklists are distributed to the team members. Individually, each team member investigates the document in order to understand it but not to detect defects. In the following phase, *Inspection Meeting*, the document is paraphrased by the reader. During this process, the inspectors can raise issues regarding the document. If the team agrees that the issue is a defect, it is classified as *missing*, *wrong*, or *extra*. Its severity is also classified as either *major* or *minor*. The defect will then be recorded by the *recorder*. The meeting moves on until the inspection team finishes inspecting the document or within a time limit of not more than two hours. Further, the moderator will hand over the defect list to the author who makes the necessary corrections. This phase is usually called *rework*. In the following phase, *follow-up*, the moderator ensures that all required changes have been made. Based on the inspection results, the moderator usually decides whether or not a re-inspection is required.

Fagan's model has placed a great emphasis on meeting. It is only during the inspection meeting, does the process of defect detection and reporting actually take place. Inspection practitioners attribute the great findings in the inspection meeting to "synergy" which generally means "combined action or operation". Based on Fagan's model, there were several inspection models developed. These models, however, also require every inspector to be at the same place and at the same time for the meeting, which is generally called synchronous meetings. These include Humphrey's model [5], Gilb and Graham inspection model [6] and N-Fold inspection model [7]. These models were implemented and many benefits were reported. However, the costs of implementing such models have been reported to be very high and some other trends have appeared in order to decrease such costs.

3.0 FOLLOW-UP STUDIES

The question of whether or not inspection meeting offsets its cost has been controversial in this decade. Many articles, questioning the meeting value, were published such as "Does every inspection need a meeting" by Votta [8] and "Does ever inspection really need a meeting" by Johnson [9]. Some researchers have anticipated that asynchrony will replace the inspection meeting in the future [10]. Some others reached the conclusion that the inspection meeting is of doubtful value. Many experiments were conducted to check that doubtful value.

Votta at AT&T Bell Labs [8] observed a series of inspection meetings involving software professionals working upon industrial projects. His analysis from the data gathered suggest that the number of defects found in meetings is only 4% greater than the number discovered during individual preparation. He also conducted subsequent cost-benefit study to compare meeting based inspections with a process based around individual defect depositions. He reported that the potential benefit of finding more defects in a meeting was not adequately offset by the higher cost incurred in organizing the meeting. Votta also noticed that only two of the inspectors can interact in the meeting at any one time. Straightforwardly, he concluded that around 30%-80% of other inspectors' time was spent listening to the current conversation.

McCarthy et al. [11] has conducted a series of experiments to investigate the notion that the inspection meeting is responsible for uncovering many defects and the effectiveness of the meeting in finding defects compared with other defect detection techniques. They utilized three detection techniques for testing the hypothesis namely Preparation – Inspection (PI), Detection – Collection (DC) and Detection – Detection (DD). PI is a technique whereby the inspectors at first try to understand or only survey the

document and only looks for defects later in the meeting. DC is a technique where inspectors go through the document to find defects during the meeting. The individual inspector simply reports his/her own findings. DD is a technique where another round of checking is added after the initial individual checking. Among the three methods used, DD was found to be the best. The DD technique recorded a detection rate of 46% followed by DC (23%) and finally PI (19%). Upon this finding, they concluded that meetings are not necessarily vital to successful inspections. However, they reported that further study is needed to confirm this finding.

Porter et al. [12] have come to a conclusion that inspection meeting gains is approximately zero. In a series of experiments, they compared the number of defects found for the first time at the meeting (meeting gains) and the number of the defects found before the inspection meeting by any of the inspectors that have not been found in the meeting. They found that there is no significant difference between the two findings. Similar results have been concluded by the replication of the same experiment both in Italy by Lanubil and Vissagio [13] and in UK by Miller et al [14].

In another study, Porter and Johnson [15] compared two experimental studies of software review meetings. The experiments compared the performance of two different groups performing inspections: "real" inspection groups, which are involved in the normal inspection meeting, and "nominal" inspection groups, in which the result of the inspection is the correlation of individual inspectors' results. They performed the comparison to test five hypotheses under the context that the real groups will outperform the nominal group. However, the studies failed to discover any significant difference in the number of defects found by the two groups. Instead, they found that the number of issues produced by the nominal groups were significantly more than the ones produced by the real groups. This has raised a doubt about the effectiveness of the inspection meeting. On the other hand, these studies have revealed that the group meeting is more effective than a meeting-less inspection in identifying the false positive defects. Also, inspection meeting has been identified to be more effective in finding some certain types of defects. They conclude that the inspection meeting does not in itself increase nor does it decrease the detection capability of the inspection process.

Land et al. [16] have confirmed the findings obtained by Porter. They found that the number of new defects reported by interacting groups (IG), groups that interact during the inspection meeting is low. According to this result, they discounted the synergy to be the important factor behind the inspection meeting. They also reported that there are defects discovered by individuals that have not been found by groups. The most important point they concluded is that interacting groups are the best at the discrimination between true and false positive defects. Therefore, they still have the performance advantage over the nominal groups in terms of the net defects. Land et al. [17] have reconfirmed these results and have demonstrated that interacting groups are the preferred choice over the average individuals and the nominal groups.

Finally, based on the above arguments, Glass has concluded that inspection meeting is of doubtful value [2].

The above results have resulted in development of new models. These models have reduced the load on the inspection meeting by distributing the process and discharging it from the inspection processs.

4.0 ADVANCED WORK

Some asynchronous inspection models have been developed. The main concern of these models was to practice the software inspection without the need for all the inspectors to be present at the same time or to convene at the same place.

4.1 Formal Technical Asynchronous Review Method

The model was developed by Philip Johnson [18]. It has three identified roles: *moderator, producer* and *reviewer*. The moderator is the person who is in charge of the overall process, the producer is the author of the document and the reviewer is the person who performs the checking. The process goes through seven phases: *setup*, *orientation*, *private review*, *public review*, *consolidation*, *group review meeting* and *conclusion*. In the *setup* phase, the inspection team is identified and the work product is put available using a computer tool called CSRS [19]. In the following phase, *orientation*, the inspection team is briefed about the inspection materials and objectives. In the *private review*, reviewers check the document and create annotations. In this phase, the annotations are kept private. However, they publicly become available in the *public review* phase. Reviewers can view all comments and also add comments. New annotations can be added at this phase as well. When the team resolves all issues, or the moderator decides to terminate the discussion, this phase is considered complete. The *consolidation* phase then follows in which the moderator analyses the results of the private and public review phases, and summarises unresolved issues. Based on the results, the moderator will decide whether or not a *group reviews meeting* will take place. The final phase is the *conclusion* where the moderator produces the final inspection report and the inspection metrics reports. The process is shown in Fig. 2.

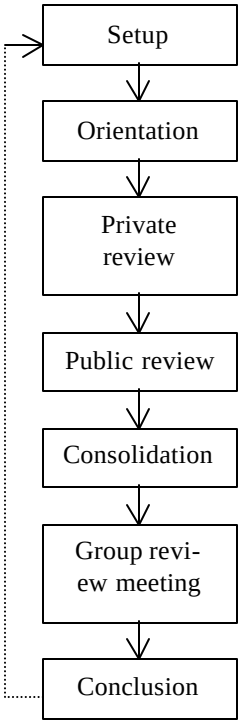

Fig. 2: FTArm Model

4.2 Mashayekhi et al Asynchronous Inspection Model

The second model was introduced by Mashayekhi *et al*. [20]. The process has been introduced to build three computer inspection tools. Unlike the first model, this model makes use of Humphrey's inspection process [5]. The process goes through the same phases except the inspection meeting, which has been substituted by a sequence of defect discussions. In some cases synchronous inspection meeting is held to resolve issues that have not been resolved asynchronously.

The process starts with the *initialisation* phase where the moderator makes the inspection materials available to the whole inspection team. Afterwards, the reviewers start to check the work product in what is called *fault collection*. A fault list is usually produced by each reviewer. In the following phase, *correlation*, the producer correlates the fault lists in one list. The correlated list is then posted to the inspection team for further asynchronous discussion. This phase is usually called *asynchronous meeting.* If the team manages to resolve all the issues in the correlated fault list, an action-item list with the resolved issues and suggested resolutions is forwarded to the producer for *rework*. On the other hand, if the team failed to resolve some issues, then the moderator decides whether or not a synchronous meeting phase should take place. The action-item list is passed to the author to make the necessary corrections. This phase is called *rework*. The moderator, in the traditional *follow-up* phase, assures that all the changes have been properly made.

4.3 Murphy and Miller's Asynchronous Inspection Model

The third model was introduced by Paul Murphy and James Miller [21, 22]. The process replaces the inspection meeting with another round of individual inspection. As shown in Fig. 3, the process starts with the *planning* phase in which the moderator prepares the ground for the inspection. The first round of *individual preparation* then starts. Here, inspectors individually go through the document looking for defects. When reaching a pre-stated deadline, each inspector circulates his/her own defect list to the rest of the team and the moderator for review. Using a communication mechanism such as email, inspectors then discuss those defects. A second round of *individual review* is followed. Learning from others inspectors' defects, an inspector can generate new defects, reclassify or delete his/her old ones. The outcome of the second round is then submitted to the moderator who collates the defect lists into one list. This list is sent to the author for rework. In contrast to Humphrey's inspection process, the model opposes the idea that the author can participate in the defect detection or collation. The final phase is the traditional follow-up activities.

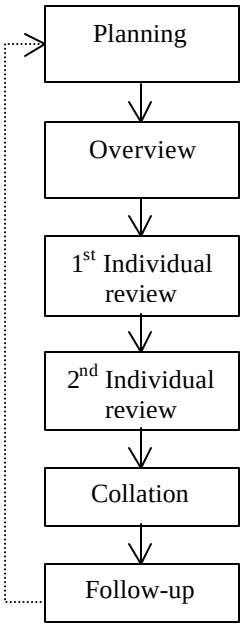

Fig. 3: Murphy and Miller's Asynchronous Inspection Model

4.4 Summary

The above three models have disposed the synchronous inspection meeting. Two models namely FTArm and Mashayekhi's models have retained it in the final stages based on the result of the asynchronous discussions. FTArm and Murphy's models rely on the consolidation phase to reach the final action list. They never allow any asynchronous discussion nor do they get the approval from

the whole group for the final list of defects. On the other hand, Mashayekhi's model allows the asynchronous discussion to take place on the correlated list allowing the inspection team to reach consensus on the final list. Mashayekhi, however, stated that 15% of the defects were passed to the synchronous meeting and that they were all categorized major [23]. This conveys that 75% out of the whole defects passed to the asynchronous meeting were minor defects. We believe that if only the 15% of the defects were passed to the asynchronous meeting there would be a better result. In the light of this, we introduce one step before the asynchronous meeting where minor defects are approved by the inspection team and the major defects are passed to the asynchronous discussion.

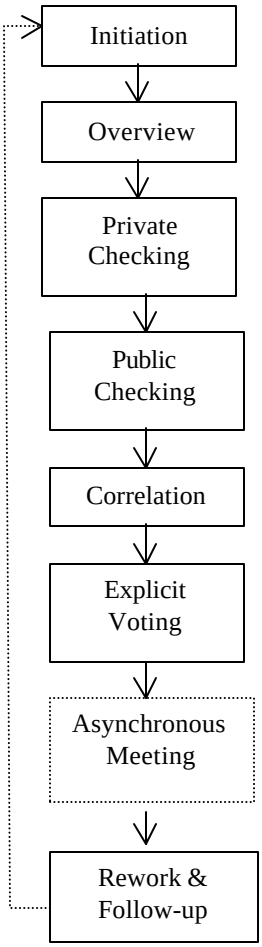

Fig. 4: Asynchronous Inspection Model

5.0 ASYNCHRONOUS INSPECTION MODEL

5.1 The Model

From the previous sections, we can generally agree that there are strong arguments to support the inspection process without meetings. An attempt is made here to introduce the inspection model which is aimed to displace the inspection meeting but allow the interaction among the group to be carried out asynchronously. In order to increase the interaction among the group and to make it centred on solving major defects, the model introduces "explicit voting" where defects are either approved or disapproved by inspectors. Simple defects can be passed directly to the action list and only debatable defects are passed to the asynchronous meeting. This saves inspectors' time in reaching consensus regarding minor defects. The model goes through the following steps:

1. Initiation

The moderator plans the inspection. He/she assures that the work product meets the entry criteria, selects the participants, prepares the inspection materials, identifies the inspection time slots, and notifies the participants.

2. Overview

A copy of the work product with any other supporting documents such as source document, standards and checklist are made available to all participants. Optionally and upon the moderator's request, the author briefs the inspection team about the work product.

3. Private Checking

Inspectors check the document individually. They write down defects with their classifications, positions and the respective time taken to find each defect. Inspectors are not allowed to share their thoughts at this stage. After a pre-specified date, this phase is declared over. Upon the progress of inspectors, the moderator can also declare this phase over at any time. The result of this stage is inspectors' defect lists.

4. Public Checking

This phase is similar to the previous one with a slight different action where inspectors share comments on the same document. A copy of all defects is distributed among inspectors. Inspectors revise these defects and add, delete, or modify their old ones. Once this phase is declared over, each inspector has to pass a final defect list to the author.

5. Correlation

The author correlates the different defect lists generated by inspectors. He/she groups similar defects and passes unique ones to the correlated list. He/she is not allowed to alter the status of such defects. One step the author has to do is to propose actions to rectify each defect in the correlated list. Once the author finishes with this phase, he/she passes the correlated list to the moderator who accordingly notifies other participants to participate in the next phase.

Although there are some opposition for the authors to take over the correlation [21], it has been decided to grant the authors this responsibility. This choice was made on the

grounds that they are the only ones who possess greatest understanding of the work product. They accordingly are the best personnel who can efficiently correlate defects. Moreover, the model confines the authors within correlating similar defects only. They are not allowed to alter any unique defects, nor are they allowed to remove any defects. In addition, knowing that this list will undergo additional investigations by the whole team in the following phases keeps the authors doing their job in the right track.

6. Explicit Voting

Inspectors vote for each defect in the correlated list. Each inspector either accepts the defect and the action that is going to be taken or rejects it. Inspectors pass their polls to the moderator who accordingly split the correlated defect list out into two lists: accepted defects and rejected defects. A defect is considered accepted if it is accepted by the majority of inspectors. It is considered rejected in any other case. At this stage the moderator passes the accepted defects to the action list and optionally, upon the result, calls for an asynchronous meeting for further discussion regarding rejected defects.

Despite the fact that group approval might have been implicitly done during the public checking phase, this stage is added to explicitly filter out uncontroversial defects. Thus, it reduces the agenda of the next phase to be more confined within rejected and usually difficult defects.

7. Asynchronous Meeting

Inspectors attend concurrent discussion regarding defects. This can be done by using a communication mechanism such as electronic mail or computer tools. Once the inspectors reach a consensus regarding a defect, it is considered resolved. The moderator is responsible for passing resolved defects to the action list. He/she is also responsible of resolving any deadlock discussions. Such defects have to be passed to the unresolved list for further actions.

8. Rework and Follow-up

The author corrects the defects found and passes the document to the moderator who confirms the corrections. Upon the inspection result, the moderator either calls for another inspection or declares the inspection complete. The moderator issues a report summarizing the inspection result and notifies the inspection team about his decision.

5.2 The Roles or the People Involved

There are three main roles in this inspection process:

- The moderator - the person who is in charge of conducting the whole process. He/she plans, initiates, controls and reports the inspection results. He/she is responsible for selecting inspectors, notifying them, controlling the moving from one phase to another and finally handling over the defects to the author for corrections. As this model has to be supported by a medium of communications, the moderator is responsible for monitoring the communications among the inspection team.

- The inspector - the person whose main task is to detect defect. They have to check the document privately at first and then publicly. They also need to vote for correlated defects. Finally, they are involved in concurrent discussions regarding rejected defects.

- The author - the person whose work is being inspected. He/she has to brief the inspection team about the inspection materials and clarify any ambiguity if there exists any. He/she is also the one who has to correlate defects and later makes the necessary corrections in the rework process.

5.3 The Documents

There are several documents which are used in this process model. We summarize the documents produced at each phase in the following table.

Table 1: Summary of documents used in the process

Phase	Documents used	Documents produced
Initiation		
Overview	Work product Source document Standards Checklists	
Private Checking	Work product Source document Standards Checklists	Individual defects lists
Public Checking	Work product Source document Standards Checklists Individual defects lists	Individual defects lists
Correlation	Individual defects lists	Correlated defects list
Explicit Voting	Work product Source document Standards Correlated defect list	Action list Rejected defects list
Asynchronous Meeting	Work product Source document Standards Rejected defects list	Action list Unresolved defects list
Rework & Follow-up	Work product Action list Unresolved defects list	Inspection report

6.0 TOOL SUPPORT

There are many existing on-line tools, which have been developed to support software inspection process. For example, ICICLE [23] (Intelligent Code Inspection in a C Language Environment), Scrutiny [24] which is a collaborative tool that supports inspecting software life cycle products and InspeQ (Inspecting software phases to ensure Quality) [25]. Since the explosion of the web, many of the inspection tools developed are based on the web. Examples of web-based tools are AISA [3], hyperCode [26] and WiP [27].

In conjunction with the existing tools, we have developed a web-based tool to support the above proposed model. The tool is called WASIT (a Web-based Asynchronous Software Inspection Tool). We have selected World Wide Web as a medium of communication in building WASIT because it offers several advantages over the other available communication mediums such as highly platform-independent, more global and no accessing time limits.

WASIT enables the whole inspection process to be automated. Usually, the process starts with the moderator setting up the documents and participants in the *initiation phase*. The moderator needs to fill in the details of the whole team. This usually includes names, handles, e-mail addresses and passwords. He/she also specifies the inspection documents and sets them up. At the end of initiation phase, WASIT generates an automatic notification through the electronic mail notifying each participant with his/her user name, password and WASIT web address. Inspection participants then can log in to the tool using the pre-mentioned web address. In the *private checking phase*, inspectors can view the document under inspection using the main document browser and record any defects they find (Fig. 5). Each inspector is confined within viewing, deleting or updating his/her own defects. However, both the moderator and the author can view all the defects. The moderator can assess the progress of each inspector using the server side or through the automatic notifications that he/she receives when an inspector finishes this phase. Upon the moderator's decision, the process can be pushed forward to the *public checking phase*. In this phase, inspectors will again go through the main documents looking for new defects. In this phase, the whole team can view the whole list of defects. Inspectors can still change or delete their own defects. They can also add new ones. Once the moderator decides that the next phase shall start, he/she can either use the server or the client side to proceed to the correlation phase. Using the correlation browser, the author then correlates the defect lists submitted by the inspectors. This phase usually integrates the different defect lists into one list. New correlated defects are usually added with an action to rectify them. As soon as the author finishes correlating defects, WASIT notifies the moderator who can move the process to the explicit voting phase. Inspectors usually receive a message in their electronic mail box stating that this phase has started. They view the correlated defect list and vote for each defect in it by using the voting browser. The voting is either 'accept' or 'reject'. Once the last inspector votes on the list, WASIT concludes the voting result. Accepted defects are usually passed to the resolved list and rejected defects are passed to the rejected defect list. Based on the result of this phase, the moderator either moves the process to the asynchronous inspection meeting or reports the inspection results declaring the process over. In the former case, the inspection team involves in parallel discussion regarding rejected defects. They can post messages to each defect discussion. These messages are usually ordered in a chronological order. They can also propose solutions for any particular defect. WASIT allows one proposal to be attached to any defect. Only inspectors are allowed to vote for proposals. The moderator is granted the right to end any discussions and to finish this phase. Resolved defects are usually passed to the action list. In the following phase, rework and follow-up, the moderator generates a list of resolved defects and passes them to the author for corrections. Using WASIT and based on the inspection result, the moderator can take the right action regarding the whole process. He/She can report the inspection process and notify the whole team using WASIT.

7.0 INITIAL EVALUATION

The above model and the prototype tool have been initially evaluated at the Faculty of Computer Science and Information Technology, University of Malaya. Two groups were selected to perform the inspection based on the proposed model and using the implemented tool. Each group has four graduate students doing Master and Ph.D. programs at the faculty. The two groups inspected some C++ documents and reported most of the defects that had been seeded within the document. Most of the defects were agreed on at the explicit voting phase (before the asynchronous meeting took place). This is due to the simplicity of the work products selected. The code defects are usually more evident than any other defects, for instance, user requirements documents' defects.

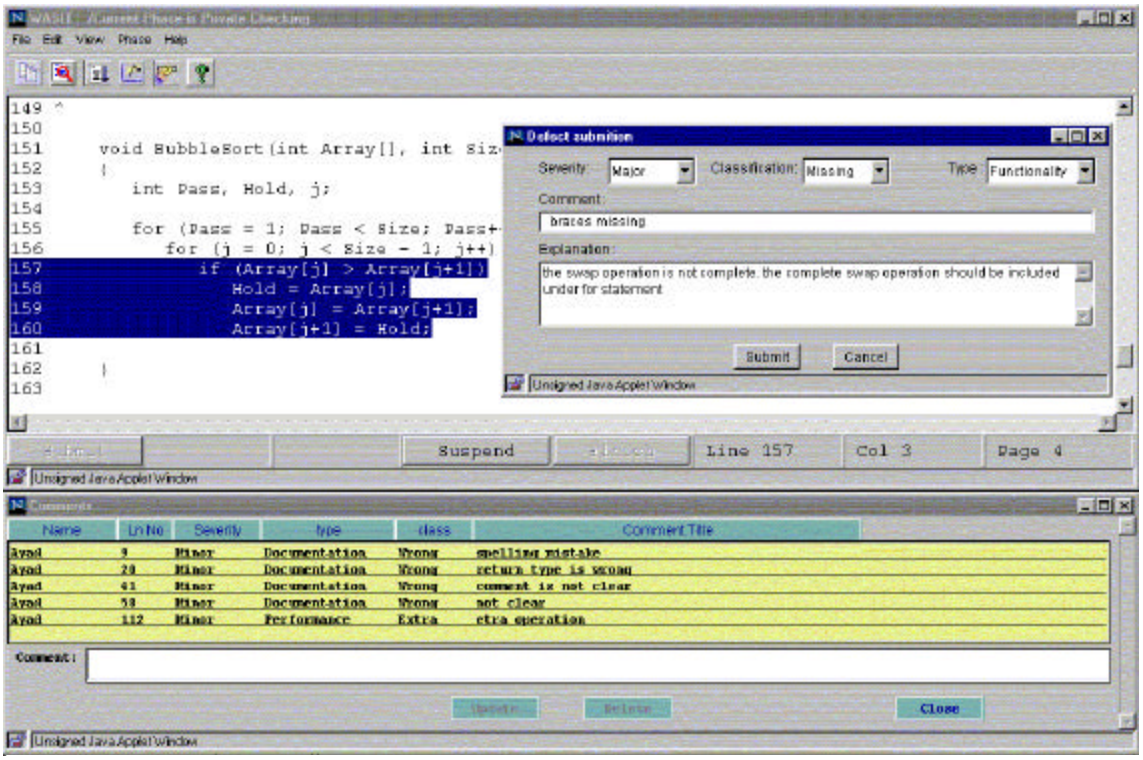

Fig. 5: WASIT defect submission at the private checking phase

8.0 FURTHER WORK

Further evaluations are needed to investigate the validity of the proposed model. Many experiments are needed to validate and compare the results with other asynchronous models. This requires some changes in the tool to accommodate other models and perform the inspection with the same materials using the different models.

9.0 CONCLUSION

This paper has introduced another asynchronous inspection model. The model has disposed the inspection meeting and introduced an additional step before the asynchronous meeting. The model has been supported by a web-based computer tool. Both the inspection model and the implemented prototype have been initially evaluated. Further evaluations are going on to validate the efficacy of the introduced model.

REFERENCES

[1] M. E. Fagan, Design and Code Inspections to Reduce Errors in Program Development. *IBM Systems Journal*, 15(3):182–211, 1976.

[2] R. L. Glass, Inspections-Some Surprising Findings, *Communications of the ACM*, 42(4):17–19, April 1999.

[3] V. Mashayekhi, *Distribution and Asynchrony in Software Engineering*. Ph.D. thesis, University of Minnesota, March 1995.

[4] M. E. Fagan, Advances in Software Inspection. *IEEE Transactions on Software Engineering*, 12(7): 744–751, July 1986.

[5] W. S. Humphrey, *Managing the Software Process*, Chapter 10, pp. 171–190. *Addison-Wesley*, 1989.

[6] T. Gilb and D. Graham, Software Inspection. *Addison-Wesley*, 1993.

[7] J. Martin and W. T. Tsai (1990). N-Fold Inspection: A Requirements Analysis Technique. *Communi-cations of the ACM*, 33(2):225–232, February 1990.

[8] L. G. Votta, Does Every Inspection Need a Meeting? In *Proceedings of the First ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pp. 107–114, December 1993.

[9] P. M. Johnson and D. Tjahjono, Does Every Inspection Really Need a meeting? *Journal of Empirical Software Engineering*, Volume 4, Number 1, January, 1998.

[10] P. M. Johnson, *Reengineering Inspection: The Future of Formal Technical Review*, *Communications of the ACM*, 41(2):40-52, February, 1998.

[11] P. McCarthy, A. A. Porter, H. Siy, and L. Votta, An Experiment to Assess Cost Benefits of Inspection Meetings and Their Alternatives. Technical Report, Computer Science Dept., University of Maryland, 1995.

[12] A. A. Porter, L. Votta, and V. Basili, Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment. *IEEE Transactions on Software Engineering*, 21(6):563{575, 1995.

[13] F. Lanubile and G. Visaggio, Assessing Defect Detection Methods for Software Requirements Inspections Through External Replication. Technical Report, Dept. of Informatica, University of Bari, 1996.

[14] J. Miller, M. Wood, M. Roper, and A. Brooks, Further Experiences with Scenarios and Checklists. Technical Report, Dept. of Computer Science, University of Strathclyde, 1996.

[15] A. A. Porter and P. M. Johnson, Assessing Software Review Meeting: Results of Comparative Analysis of Two Experimental Studies. *IEEE Transactions on Software Engineering*, 23(3):129–145, March 1997.

[16] L. Land, C. Sauer, and R. Jeffery, Validating the Defect Detection Performance Advantage of Group Designs for Software Reviews: Report of a Laboratory Experiment Using Program Code. In Jazayeri M. and Schauer H., editors, *Sixth European Software Engineering Conference Held Jointly with the Fifth ACM SIGSOFT Symposium on Foundations of Software Engineering*, Number LNCS 1301 in Lecture Notes in Computer Science, pp. 295-309. Springer, September 1997.

[17] L. Land, R. Jeffery and C. Sauer, Validating the Defect Detection Performance Advantage of Group Designs for Software Reviews: Report of Replicated Experiment. In Bailes P. A., editor, *Proceedings of Australian Software Engineering Conference, IEEE* Computer society, pp. 17-26, October 1997.

[18] P. M. Johnson, An Instrumented Approach to Improving Software Quality Through Formal Technical Review. In *Proceedings of the 16th International Conference on Software Engineering*, May 1994.

[19] P. M. Johnson and D. Tjahjono (1993). CSRS Users Guide. Technical Report ICS-TR-93-16, Collaborative Software Development Laboratory, Department of Information and Computer Sciences, University of Hawaii, 1993.

[20] V. Mashayekhi, C. Feulner, and J. Reidl, CAIS: Collaborative Asynchronous Inspection of Software. In *Proceedings of the Second ACM SIGSOFT Symposium on the Foundations of Software Engineering*, December 1994.

[21] P. Murphy and J. Miller, Asynchronous Software Inspection, Technical Report EfoCS-27-97, Department of Computer Science, University of Strathclyde, 1997.

[22] P. Murphy and J. Miller, A Process for Asynchronous Software Inspection. In *Proceedings of the 8th International Workshop on Software Technology and Engineering Practice*, pp. 96–104, July 1997.

[23] L. R. Brothers, V. Sembugamoorthy and M. Muller, ICICLE: Groupware for Code Inspections. In *Proceedings of the 1990 ACM Conference on Computer Supported Cooperative Work*, pp. 169–181, October 1990.

[24] J. W. Gintell, J. Arnold, M. Houde, J. Kruszelnicki, R. McKenney and G. Memmi, Scrutiny: A Collaborative Inspection and Review System. In *Proceedings of the Fourth European Software Engineering Conference*, September 1993.

[25] E. A. Meyers, and J. C. Knight (1992). An Improved Software Inspection Technique and an Empirical Evaluation of its Effectiveness. Technical Report TR-92-15, Department of Computer Science, University of Virginia, May 1992.

[26] J. M. Perpich, D. E. Perry, A. A. Porter, L. G. Votta and M. W. Wade (1997). Anywhere Anytime Code Inspections: Using the Web to Remove Bottlenecks in Large-Scale Software Development. In *Proceedings of the 19th International Conference on Software Engineering*, pp. 14–21, 1997.

[27] L. Harjumaa and I. Tervonen, A WWW-Based Tool for Software Inspection. In *Proceedings of HICSS-98*, Volume III, pp. 379–388, 1998.

BIOGRAPHY

Abdusalam is a graduate student in Faculty of Computer Science & Information Technology, University of Malaya. He is pursuing his Master in Software Engineering and currently developing a web-based tool for software inspection. His research areas include software metric and software technology. He is also an affiliate member of ACM and IEEE Computer Society.

Rodina Ahmad obtained his Master of Computer Science from Rensselaer Polytechnic Institute (USA) in 1991. Currently, she is a lecturer at the Faculty of Computer Science and Information Technology, University of Malaya. Her research areas include object-oriented software developments, software technology and information systems. She has published a number of papers related to these areas. She is a member of ACM.