APOIO COMPUTACIONAL PARA INSPEÇÃO DE SOFTWARE

KATIA ROMERO FELIZARDO
UNOESTE - Universidade do Oeste Paulista
FIPP – Faculdade de Informática de Presidente Prudente - Departamento de Computação
Rua José Bongiovani, 700 – Presidente Prudente/SP - CEP 19050-900
kafelizardo@apec.unoeste.br

Resumo: Este artigo tem como objetivo apresentar as características gerais de algumas ferramentas que foram identificadas na literatura e que apóiam o processo de inspeção [Fagan, 1986], assim como discutir algumas vantagens e desvantagens da automatização nas atividades de inspeção.

Abstract: The purpose of this paper is to present the general features of some tools identified in the literature that support the inspection process [Fagan, 1986], and to discuss some advantages and disadvantages of the computational support during the inspection activity.

Palavras-Chave: Engenharia de Software, Inspeção de Software, Apoio Automatizado, Experimentação.

1 INTRODUÇÃO

A inspeção é um processo de revisão formal de software e corresponde a uma das mais importantes atividades de *GQS – Garantia de Qualidade de Software*, sendo que o principal objetivo é a descoberta antecipada de falhas (produção de uma saída incorreta em relação à especificação), de modo que eles não se propaguem para o passo seguinte do processo de software. Assim, a Engenharia de Software tem utilizado a inspeção como um dos métodos mais eficientes e efetivos na busca por um produto de melhor qualidade.

Durante a condução de uma RTF - Revisão Técnica Formal (inspeção) - várias informações devem ser coletadas, desde as informações sobre os artefatos até aquelas referentes à coleta dos dados decorrentes da própria aplicação da revisão. Muitas dessas atividades podem ser automatizadas, sendo assim, torna-se muito importante o uso das ferramentas de apoio para facilitar a atividade de inspeção.

2 O PROCESSO DE INSPEÇÃO

O processo de inspeção foi descrito primeiramente por Michael Fagan e é composto por seis fases, que são: *Planejamento, Apresentação, Preparação, Reunião de Inspeção, Retrabalho e Acompanhamento* [Fagan, 1986].

No *Planejamento* os inspetores são selecionados e os materiais a serem revisados são preparados. Na *Apresentação,* o grupo recebe instruções essenciais sobre o material a ser inspecionado, especialmente sobre o que deve ser inspecionado. Na *Preparação,* integrantes do time de inspeção se preparam para desempenhar o papel designado a cada um. No momento da *Reunião de Inspeção* os defeitos são encontrados, discutidos e categorizados. No *Retrabalho* o autor do documento corrige os defeitos encontrados pelo time de inspeção. Na etapa de *Acompanhamento,* o time de inspeção é responsável por assegurar que todos os defeitos encontrados foram corrigidos e nenhum outro tipo de defeito foi introduzido na fase de *Retrabalho*. O *Acompanhamento* também pode ser realizado somente pelo moderador [MacDonald et al, 1995a] [Fagan, 1986].

Um grupo de inspeção envolve desenvolvedores de software, entre outros participantes, em um processo formal de investigação. Consiste de três a oito participantes e inclui os seguintes papéis: o *Autor*, o *Moderador*, o *Redator* e o *Inspetor*.

O *Autor* é o desenvolvedor do produto a ser inspecionado. O *Moderador* é o membro da equipe que lidera a inspeção, programa e controla as reuniões. O *Redator* é aquele que tem como função relatar os defeitos (passo, processo ou definição de dados incorretos, como por exemplo, uma instrução ou comando incorreto) e as questões surgidas durante a inspeção. O *Inspetor* é o papel assumido por cada integrante do time de inspeção que tenta encontrar

defeitos no produto, sendo que todos os participantes podem agir como inspetores além de executarem outras atribuições.

A inspeção pode ser executada ao final de cada fase do desenvolvimento do software. Exemplos de técnicas que podem ser utilizadas são as técnicas de leitura. Estas técnicas são formadas por um conjunto de instruções passadas para o leitor com o objetivo de informá-lo de como ele deve ler e o que olhar em um produto de software. As técnicas de leitura objetivam realizar uma tarefa particular como, por exemplo, detectar defeitos através de uma análise individual de um produto de software, o qual pode ser tanto uma especificação de requisitos, como um projeto, um código ou mesmo um plano de teste [Basili et al, 1996].

Para inspecionar os requisitos são utilizadas técnicas como o PBR (Perspective-Based Reading) [Basili et al, 1996] [Basili & Shull, 2000].

Na seqüência serão descritas as características gerais de ferramentas de apoio computacional ao Processo de Inspeção de Software.

3 Características Gerais de Ferramentas para Inspeção de Software

MacDonald et al [MacDonald et al, 1995a] [MacDonald et al, 1995b] e MacDonald & Miller [MacDonald & Miller, 1996] [MacDonald & Miller, 1997] discutem alguns aspectos de automatização de experimentos para o processo de inspeção, descrito por Fagan [Fagan, 1986]. Nesse contexto, eles identificaram quatro atividades principais que são possíveis de serem automatizadas: *Manipulação de Documentos; Preparação Individual; Reunião de Apoio* e *Coleta de Dados.*

Na *Manipulação de Documentos* uma das atividades que pode ser auxiliada por ferramenta, segundo os autores, é a disponibilização dos artefatos usados na inspeção de maneira *on-line*. Além disso, com o apoio de ferramenta é possível tratar diferentes tipos de documentos e disponibilizar suporte de anotação. Assim, é possível, que as anotações sejam associadas com a parte do documento à qual fazem referência. As anotações podem permitir, por exemplo, o registro do tipo do defeito e de sua descrição.

Na *Preparação Individual* uma ferramenta pode indicar aos inspetores os defeitos mais simples, disponibilizar *checklists* e outros documentos de apoio de maneira *on-line.* Ainda durante essa atividade uma ferramenta pode garantir que cada item do *checklist* seja percorrido.

Na *Reunião de Apoio* uma ferramenta pode, por exemplo, verificar a quantidade de tempo gasto por cada inspetor na *Preparação Individual.* O objetivo desse controle é liberar os inspetores que tenham tomado contato com os documentos durante um período considerado adequado antes de participarem da reunião. Uma ferramenta pode também permitir uma reunião distribuída na qual os participantes não estejam necessariamente numa mesma sala. A reunião distribuída pode ser conduzida com algum suporte de reunião eletrônica, como detalhado em [Nunamaker et al, 1991].

Na etapa de *Coleta de Dados* uma ferramenta pode permitir que os dados da inspeção sejam coletados automaticamente e que a análise dos mesmos seja realizada de maneira automatizada. A ferramenta pode ser capaz da detecção automática de defeitos, através da identificação dos defeitos mais simples e geração automática de anotações sobre o defeito para que o mesmo possa ser revisado pelo inspetor.

Durante os vários passos do processo de inspeção, os participantes precisam registrar vários tipos de dados, principalmente aqueles relativos aos defeitos encontrados. Em geral, vários formulários em papel são preenchidos para depois essas informações serem armazenadas em um banco de dados para posterior análise também manual.

Assim, para Laitenberg & Dreyer [Laitenberg & Dreyer, 1998a] [Laitenberg & Dreyer, 1998b] a automatização da atividade de *Coleta de Dados* deve incluir:

- a substituição dos formulários em papel por formulários eletrônicos;
- a condução dos participantes durante a fase de *Coleta de Dados* para garantir que todos os dados necessários sejam coletados e
- o armazenamento dos dados eletronicamente, de maneira automática.

Durante a *Coleta de Dados* também podem ser coletadas informações como o número de defeitos encontrados ou o tempo total gasto na inspeção. Essas informações são aproveitadas para melhorar a avaliação sobre o processo de inspeção. Laitenberg & Dreyer [Laitenberg et al, 1998a] [Laitenberg et al, 1998b] consideram que a *Coleta de Dados da Inspeção* é essencial para monitorar, controlar e melhorar as inspeções de software.

Durante a década de 90 surgiram diversas propostas objetivando automatizar o processo de inspeção de software, MacDonald et al [MacDonald et al, 1995a] [MacDonald et al, 1995b] relacionam algumas

ferramentas que apóiam essa atividade. São elas: *ICICLE* (*Inteligent Code Inspection in a C Language Environment*) [Brothers et al, 1990] [Sembugamoorthy & Brothers, 1990], *CSI (Collaborative Software Inspection)* [Mashayekhi, 1993], *Scrutiny* [Gintell et al, 1993], *InspeQ (Inspecting software in phases to ensure Quality)* [Knight & Meyers, 1991] [Knight & Meyers, 1993] e *CSRS (Collaborative Software Review System)* [Johnson & Tjahjono, 1993].

A Tabela 1 relaciona essas ferramentas, fazendo referência aos aspectos de automatização destacados anteriormente como, por exemplo, suporte a diferentes tipos de documentos, referência cruzada, suporte de anotação, capacidade de detecção automática de defeitos, distribuição *on-line* de *checklists*, suporte à reunião distribuída, consenso de reunião e coletada de métricas.

Tabela 1 - Resumo de ferramentas de apoio à inspeção e suas características (Adaptada de [MacDonald et al, 1995a])

	ICICLE	*CSI*	*Scrutiny*	*InspeQ*	*CSRS*
Suporte a documentos	•	•	•	•	•
Referência Cruzada	•				
Suporte de anotação	•	•	•		•
Detecção automática de defeitos	•				
Checklists		•		•	•
Suporte à reunião distribuída		•	•		
Consenso			•		•
Coleta de Métricas	•	•	•		•

Em [Miller & Macdonald, 2000] os autores comentam sobre novas versões das ferramentas apresentadas na Tabela 1, mas de maneira geral, as características observadas são as mesmas das apresentadas nessa tabela.

Como mencionado anteriormente, as ferramentas citadas na Tabela 1 foram desenvolvidas com o objetivo de automatizar o processo de inspeção, descrito por Fagan [Fagan, 1986]. Atualmente, outras ferramentas que visam à automatização de técnicas de inspeção podem ser citadas, como por exemplo, uma ferramenta de apoio à aplicação das Técnicas de Leitura Baseadas em Perspectiva (PBR) [Silva, 2003], uma ferramenta de construção de casos de uso para utilização na técnica de leitura PBR e uma infra-estrutura de suporte ao processo de inspeção de software – ISPIS – [COPPE/UFRJ, 2003].

Além dessas, pode-se destacar a *COTEST (Code Testing Experiment Support Tool)* [Felizardo, 2003]. Essa ferramenta foi desenvolvida com o objetivo de apoiar a replicação de experimentos baseados em código fonte. Esse experimento explora, além de outras, uma técnica de Leitura de Código denominada Abstração Passo a Passo [Linger, 1979].

4 VANTAGENS E DESVANTAGENS DO APOIO COMPUTACIONAL

MacDonald & Miller [MacDonald & Miller, 1997] relacionam algumas vantagens e desvantagens de se utilizar o apoio computacional na atividade de inspeção:

• Vantagens:

- permite o uso dos documentos a serem revisados de maneira eletrônica, evitando a impressão de várias cópias do documento, uma para cada revisor/inspetor, reduzindo assim os custos do processo;

- as listas de defeitos e comentários produzidos durante a inspeção ficam armazenados automaticamente e

- legibilidade das listas e comentários digitados.

• Desvantagens:

- é necessário um tempo de treinamento para o aprendizado da própria ferramenta;

- a velocidade ou habilidade do participante digitar a lista de defeitos pode influenciar no número final de defeitos detectados;

- pouco espaço para distribuir as janelas da ferramenta na tela do computador, uma vez que o processo de inspeção exige que o produto a ser revisado, o *checklist* e a lista de defeitos sejam exibidos e manipulados simultaneamente. Muitos monitores não são capazes de mostrar de forma adequada três janelas ao mesmo tempo, este fato deve ser levado em consideração na etapa de elaboração e implementação de ferramentas de suporte e leitura em vídeo, uma vez que a leitura do código, dos requisitos ou das listas de defeitos deve ser feita na própria tela do computador. Segundo Dillon [Dillon, 1992] existe uma redução de 20 a 30% na velocidade de leitura de textos na tela em relação à leitura em papel, mas a compreensão não sofre nenhuma alteração, embora a leitura na tela seja mais cansativa.

Com base nas discussões apresentadas anteriormente sobre as características gerais das ferramentas para inspeção de *software* e as vantagens e desvantagens do uso desse ferramental pode-se concluir que:

- a disponibilização *on-line* dos documentos a serem inspecionados permite que os mesmos sejam distribuídos bem antes da inspeção, possibilitando aos inspetores mais tempo para leitura;

- a detecção automática dos defeitos contribui para uma redução do tempo da inspeção, que deve ser relativamente breve;

- o suporte para anotação facilita o registro formal de todas as decisões sobre a inspeção;

- o controle do tempo gasto por cada revisor na preparação individual auxilia a formação de uma equipe de inspeção que possa prestar uma contribuição realmente eficaz.

5 CONCLUSÕES

Neste artigo, foram apresentados os aspectos gerais que podem ser automatizados em um processo de inspeção segundo [MacDonald et al, 1995 a; 1995b] [MacDonald & Miller, 1996; 1997]. Foram comentadas, de maneira resumida, características de algumas ferramentas que fornecem suporte ao processo de inspeção, enfocando o quanto cada uma delas satisfaz os aspectos gerais.

Também foram relacionadas as vantagens e desvantagens da aplicação individual da inspeção baseada em papel em relação à inspeção apoiada por ferramentas.

O desenvolvimento de ferramentas de automatização contribui para redução dos custos e esforços de manipulação dos documentos a serem inspecionados. Durante a coleta e análise dos defeitos também se observa uma redução significativa.

O gasto com a correção de defeitos é maior nas fases posteriores do processo de desenvolvimento, sendo assim, as atividades de inspeção tem como principal objetivo encontrar e corrigir defeitos tão logo quanto eles sejam introduzidos. Portanto deve-se valorizar iniciativas de desenvolvimento de ferramentas que apóiam as técnicas e/ou o processo de inspeção.

Referências Bibliográficas

[Basili et al, 1996] Basili, V. R.; Green S.; Laitenberger, O.; Lanubile, F.; Shull, F.; Sorumgard, S.; Zelkowitz, M. "The Empirical Investigation of Perspective-Based Reading". Empirical Software Engineering: An International Journal, v.1, n.2, p.

[Basili & Shull, 2000] Basili, Victor R.; Shull, Forrest; RUS, Ioana. "How Perspective-Based Reading Can Improve Requirements Inspections". IEEE Computer, vol 33, nº 07. Julho, 2000.

[Brothers et al, 1990] Brothers, L.; Sembugamoorthy, V; Muller, M. "ICICLE: Groupware for Code Inspection", In Proceedings of the 1990 ACM Conference on Computer Support Cooperative Work, Outubro, 1990, pp. 169-181.

[COPPE/UFRJ,2003] COPPE/UFRJ Engenharia de Software Experimental. 2003.
Disponível em http://www.cos.ufrj.br/~ese/ (Acessado em 01/10/2003)

[Dilon, 1992] Dillon, A. "Reading from paper versus screens: a critical review of the empirical literature. Ergonomics, 35 (10): 1297 – 1326, Outubro, 1992.

[Fagan, 1986] Fagan, Michael. "Advances in Software Inspection", IEEE Transactions on Software Engineering, Vol. SE-12, NO. 7, Julho, 1986.

[Felizardo, 2003] Felizardo, Katia Romero. "COTEST - Uma Ferramenta de Apoio à Replicação de um Experimento Baseado em Código Fonte". Dissertação de Mestrado, PPG-CC UFSCar, 2003.

[Gintell et al, 1993] Gintell, J.W.; Arnold, J.; Houde, M.; Kruszelnicki, J; McKenney, R; Memmi, G. "Scrutiny: A Collaborative Inspection and Review System", In Proceedings of the Fourth European Software Engineering Conference, Grarwisch-Partenkirchen, Germany, Setembro, 1993.

[Johnson & Tjahjono, 1993] Johnson, P.M.; Tjahjono, D."CSRS User Guide" Technical Report ICS-TR-93-16, University of Hawaii.

[Knight & Meyers, 1991] Knight, J.C.; Meyers, E.A. "Phased Inspection and their Implementation", Software Engineering Notes, Vol 16, N. º 3, Julho 1991, pp 29-35.

[Knight & Meyers, 1993] Knight, J.C.; Meyers, E.A. "An Improved Inspection Technique", Communications of the ACM, Vol 11, n. º 11, Novembro 1993, pp 51-61.

[Laitenberg & Dreyer, 1998a]Laitenberg, O.; Dreyer, H.M. "Automated Software Engineering Data Collection Activities via the World Wide Web: A Tool Development Strategy applied in the Area of Software Inspection". Proceedings of the 5th. International

Symposium on Software Metrics, Institute of Electrical and Electronics Engineers, 1998.

[Laitenberg & Dreyer, 1998b] Laitenberg, O.; Dreyer, H.M. "Evaluating the Usefulness and the Ease of Use of a Web-based Inspection Data Collectionn Tool". Proceedings of the 5th. International Symposium on Software Metrics, Institute of Electrical and Electronics Engineers, 1998.

[Linger, 1979] Linger, R. C.; Mills, H. D. e Witt, B. I. "Structured Programming: Theory and Practice", Addison-Wesley, 1979.

[Lott & Rombach, 1996] Lott, C. M.; Rombach, H.D. "Repeatable Software Engineering for Comparing Defect-Detection Techniques". Journal of Empirical Software Engineering, v. 1, n. 3, p. 241-277,1996.

[MacDonald et al, 1995a] MacDonald, F; Miller, J; Brooks, A.; Roper, M; Wood, M. "Automating the Software Inspection Process", Empirical Foundations of Computer Science (EfoCS), Departament of Computer Science, University of Stranthclyde, Junho, 1995.

[MacDonald et al, 1995b] MacDonald, F; Miller, J; Brooks, A.; Roper, M; Wood, M. "A Review of Tool Support for Software Inspection", Empirical Foundations of Computer Science (EfoCS), Departament of Computer Science, University of Stranthclyde, Junho, 1995.

[MacDonald & Miller, 1996] MacDonald, F; Miller. "Automated Generic Support for Software Inspection", Empirical Foundations of Computer Science (EfoCS), Departament of Computer Science, University of Stranthclyde, Agosto, 1996.

[MacDonald & Miller, 1997] MacDonald, F; Miller. "A comparison of Tool-Based and Paper-Based Software Inspection", Empirical Foundations of Computer Science (EfoCS), Departament of Computer Science, University of Stranthclyde, Abril, 1997.

[Mashayekhi, 1993] Mashayekhi, V.; Drake, J.M.; Tsai, W.T.; Reidl, J. "Distributed Collaborative Softwae Inspection", IEEE Software, Vol. 10, N.º 5, Setembro, 1993, pp. 51-61.

[Miller & Macdonald, 2000] Miller, J.; MacDonald, F. "An empirical incremental approach to tool evaluation and improvement." The Journal of System and Software, n.51, p. 19-35, 2000

[Nunamaker et al, 1991] Nunamaker, J. F.; Dennis, A. R.; Valaich, J.S; Vogel, D.R.; George, J.F. "Eletronic Meeting Systems to Support Group Work", Communications of the ACM, Vol. 33 N.º 2, Junho, 1991.

[Sembugamoorthy & Brothers, 1990] Sembugamoorthy, V; Brothers, L. "ICICLE: Inteligent Code Inspection in a C Language Environment", The 14th Annual Computer Software and Applications Conference, Outubro, 1990, pp. 146-154.

[Silva,2003] Silva, L. F. S. Apoio Ferramental para Aplicação de Técnicas de Leitura Baseada em Perspectiva (PBR). Anais do WTES - Workshop de Teses em Engenharia de Software, 2003, p. 83–88.