Cleanroom Software Development

An Empirical Evaluation

Milena Guessi
Vinícius Augusto Tagliatti Zani

Agenda

- Introdução
- Apresentação da abordagem
- Objetivos da investigação
- Estudo empírico utilizando a abordagem
- Análise de dados e interpretação
- Conclusões

Introdução

- Necessidade de disciplina no processo de desenvolvimento de software
  - Proposta de uma abordagem: Cleanroom
- Abordagem Cleanroom visa:
  - O aumento do controle sobre o desenvolvimento;
  - Aumento na conformidade com os requisitos;
  - Maior confiabilidade do sistema construído; e
  - Maior clareza no código fonte gerado.
- Condução de um estudo empírico
  - Abordagem tradicional vs. Cleanroom

Desenvolvimento Cleanroom

- Apresentado pela IBM, divisão de Sitemas Federais
- Objetivo
  - Desenvolvimento de software com confiabilidade certificada
- Evitar defeitos na fase de desenvolvimento
- Integra:
  - Desenvolvimento iterativo;
  - Especificação formal de software;
  - Desenvolvimento não baseado em execução; e
  - Testes independentes estatisticamente executados.
- Consequentemente: menor taxa de erros, e maior confiabilidade

Desenvolvimento Cleanroom

- *Desenvolvimento iterativo*
  - O software é considerado uma sequência de incrementos executáveis em um produto
- *Métodos formais para especificação e projeto*
  - Uso de especificações estruturadas e máquinas de estado que podem ser desenvolvidas de forma incremental
- *Desenvolvimento sem execução*
  - Desenvolvedores utilizam métodos diferentes de testes para verificar a corretude de seu código, como leitura *stepwise*, inspeção de código, verificação em grupo e verificação formal

Desenvolvimento Cleanroom

- *Teste independente baseado em estatística*
  - As entradas reais do sistema são identificadas e distribuídas de maneira estatística;
  - Os casos de teste são escolhidos de forma incremental, de acordo com a evolução do projeto;
  - Podem testar também a confiabilidade além da detecção de erros;
  - Métrica para confiabilidade: tempo médio entre falhas (MTBF).

Adoção, metodologias e ferramentas

- Adoção
  - Deve haver treinamento da equipe nas tecnologias envolvidas no Cleanroom;
- Cleanroom e prototipação
  - Podem ser utilizados em conjunto, onde o segundo é utilizado na elaboração dos requisitos e depois descontinuado
- Ferramentas
  - Não são proibidas, e auxiliam no processo de revisão de código e detecção de erros. Ex: analisadores estáticos e de fluxo, verificadores de sintaxe, de tipo e formais, etc.

Objetivos de investigação

- Caracterizar o efeito de Cleanroom:
  - no produto entregue;
  - no processo de desenvolvimento; e
  - nos desenvolvedores.

Estudo empírico utilizando a abordagem

- Objetos do estudo
  - Grupos de três alunos de um curso de projeto e desenvolvimento de software;
  - Experiência balanceada (profissional – média de 1.6 anos, acadêmica e nas linguagens utilizadas)
- Projeto desenvolvido
  - Um documento de requisitos distribuído para cada equipe
    - Sistema de mensageria eletrônica
  - Prazo de seis semanas, utilizando a linguagem Simpi-T

Estudo empírico utilizando a abordagem

- Cleanroom e desenvolvimento tradicional
  - Foram realizados dois experimentos
    - 1982: desenvolvimento cleanroom – sem compilação; e
    - 1983: desenvolvimento tradicional (controle) – com compilação e demais técnicas utilizadas pela indústria.
  - Estatisticamente similares (experiência, equipes)
  - Evitou-se a “competição” entre os grupos de 82 e 83 para não haver viés
- Marcos de projeto
  - Os grupos definiram seus marcos de entrega (a cada 1-2 semanas);
  - Cada entrega foi testada independentemente com seus erros reportados para correção na próxima entrega.

Estudo empírico utilizando a abordagem

- Teste operacional
  - Um “perfil operacional” do produto foi utilizado para definir casos de teste de acordo com sua distribuição de frequência;
  - Simulação de 50 usuários de sistema
- Avaliação do projeto
  - Técnicas de desenvolvimento;
  - Resultados do teste independente; e
  - Entrevista oral final.

Análise de dados e interpretação

- Caracterização do efeito:
  - No produto desenvolvido
  - No processo de desenvolvimento
  - Nos desenvolvedores
- Distinção entre os times

Team	Cleanroom	Source Lines	Executable Statments	Procedures & Functions
A	yes	1681	813	55
B	yes	1626	717	42
C	yes	1118	573	42
D	yes	1046	477	30
E	yes	1087	624	32
F	yes	1213	440	35
G	yes	1196	581	31
H	yes	1876	550	51
I	yes	1305	608	23
J	yes	1052	658	24
a	no	824	410	26
b	no	1429	633	18
c	no	2264	999	46
d	no	1629	626	67
e	no	1310	459	43

Fig. 3: Estatísticas dos sitemas

Caracterização do efeito no produto desenvolvido

- Propriedades operacionais dos sistemas
  - Completude da implementação
    - 16 funções lógicas (mandar email para uma pessoa, ler um email, responder etc.)
    - Para cada função, atribui valor:
      - 2, se satisfez os requisitos completamente
      - 1, se satisfez os requisitos parcialmente
      - 0, se não foi disponibilizado

O nível de significância para a estatística Mann-Whitney corresponde à probabilidade do erro do tipo I em um teste unilateral

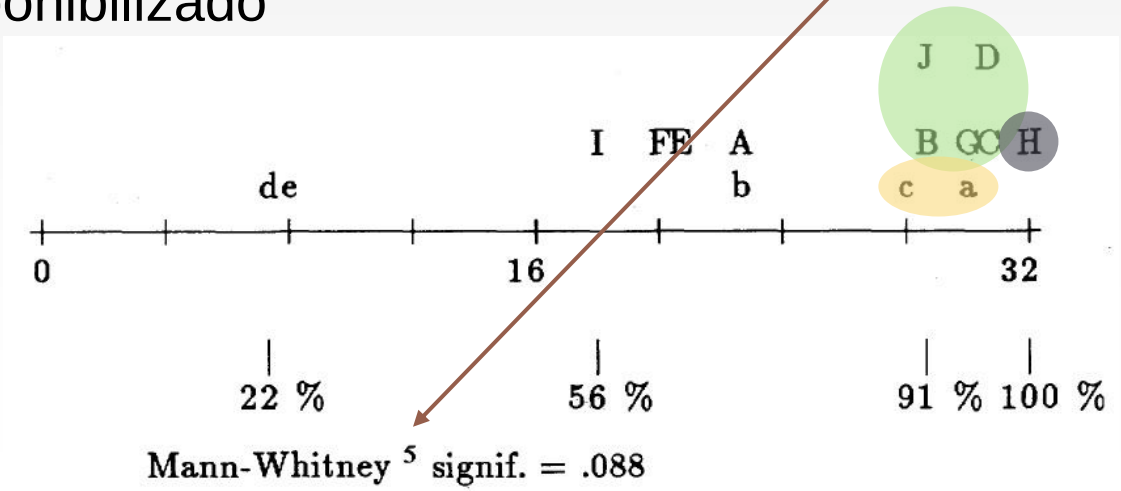


Fig. 4: Conformidade de requisitos

Caracterização do efeito no produto desenvolvido

- Propriedades operacionais dos sistemas
  - Teste operacional
    - 50 casos de teste aleatórios foram executados na versão final de cada sistema para simular o ambiente operacional
    - Se forem considerados as falhas duplicadas a melhor performance da abordagem Cleanroom não é significante
      - É preciso que a revisão de todo o sistema seja **uniforme** para que a performance do sistema não seja tão sensível ao perfil operacional

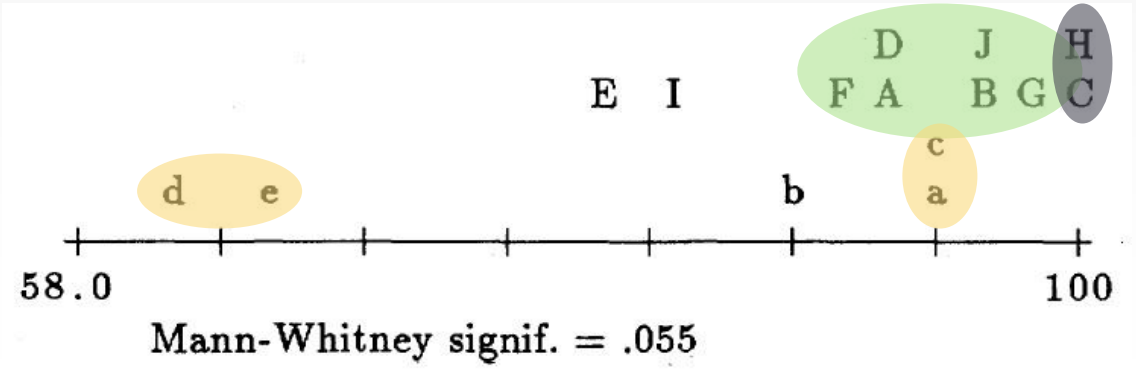


Fig. 5: Porcentagem de casos de teste bem sucedidos durante teste operacional (sem falhas duplicadas)

Caracterização do efeito no produto desenvolvido

- Propriedades operacionais dos sistemas
  - Grande variação na performance
    - Principalmente nos grupos utilizando uma abordagem mais tradicional
      - Perspectiva do testador: deixa de fora classes de funções e cometendo mais falhas únicas
    - Na abordagem Cleanroom, métodos de revisão offline são mais gerais e seu uso contribuiu para uma conformidade de requisitos mais completa e um menor número de falhas

Caracterização do efeito no produto desenvolvido

- Propriedades estáticas dos sistemas
  - Tamanho do sistema
    - Não foi observada diferença estatística entre os atributos de tamanho analisados (Fig 3)
  - Legibilidade do sistema
    - Número de comentários e *densidade de complexidade*
      - Dada pelo cálculo da complexidade sintática normalizada pelo número de declarações executáveis
    - Na abordagem Cleanroom o código foi mais comentado (MW = 0.089) e tinha menor densidade de complexidade (MW = 0.079)
  - Uso de dados
    - Na abordagem Cleanroom desenvolvedores usaram maior número de itens de dados não locais (MW = 0.071) e declarações de atribuição (MW=0.056)

Caracterização do efeito no produto desenvolvido

- Propriedades estáticas dos sistemas
  - Medidas apenas dos produtos Cleanroom relacionadas com o uso da linguagem de implementação
    - Indicam que os mais bem sucedidos desenvolvedores na abordagem Cleanroom simplificaram o uso da linguagem:
      - mais chamadas de procedimentos e declarações IF,
      - menor quantidade de declarações CASE e WHILE,
      - menor frequência de reúso de variáveis, e
      - escrita de subrotinas que requerem menor esforço de compreensão

Caracterização do efeito no produto desenvolvido

- Contribuições da experiência do programador
  - Experiência geral com linguagens de programação mostraram relação entre a porcentagem de testes operacionais bem sucedidos e completude da implementação

Caracterização do efeito no produto desenvolvido

- Resumo
  - Na abordagem Cleanroom:
    - Requisitos foram atingidos mais completamente
    - Maior porcentagem de sucesso nos casos de teste
    - Mais comentários e menor densidade de complexidade
    - Mais dados não locais e maior número de declarações de atribuição
  - Desenvolvedores mais bem sucedidos na abordagem:
    - Usaram mais chamadas de procedimentos e declarações IF
    - Usaram menos declarações CASE e WHILE
    - Reutilizaram variáveis com menor frequência
    - Desenvolveram subrotinas que necessitaram de menor esforço para compreensão
    - Possuiam maior experiência geral em linguagens de programação

Caracterização do efeito no processo de desenvolvimento

Após o desenvolvimento foi feita uma pesquisa de opinião

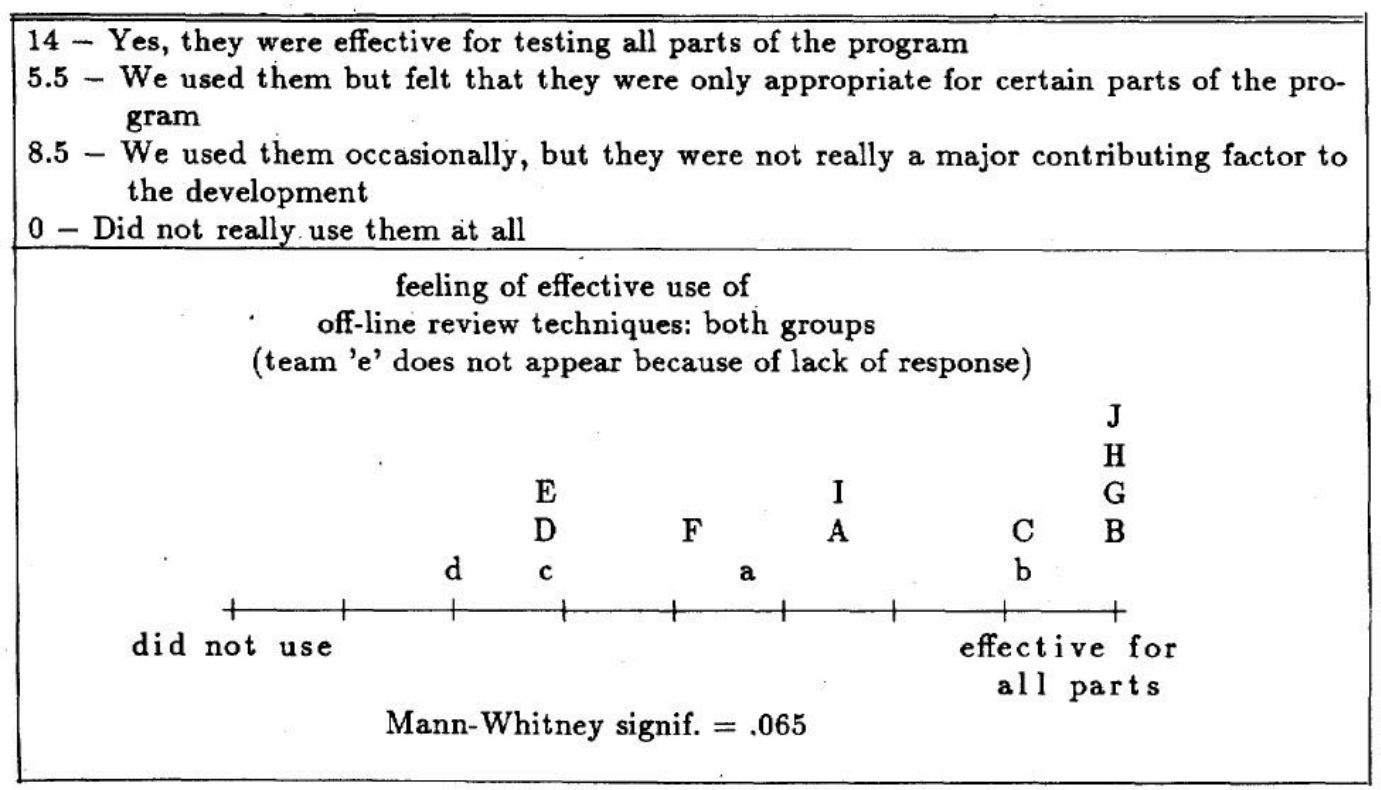
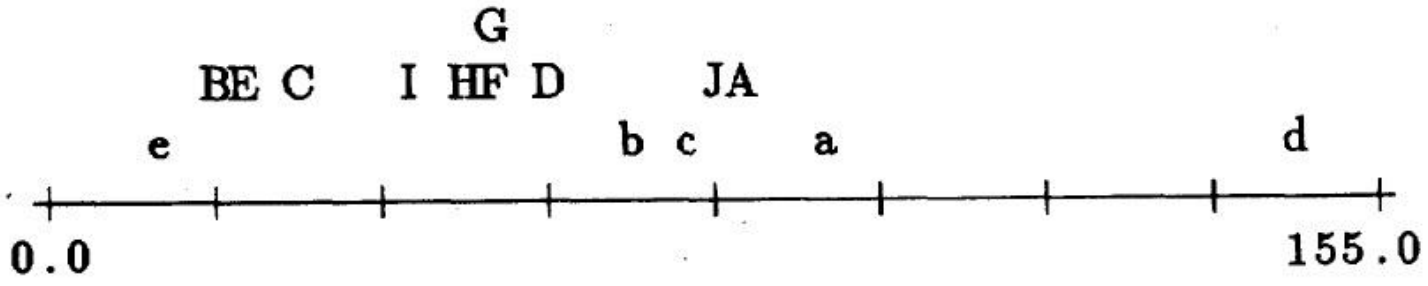


Fig. 5: Respostas obtidas para a pesquisa de opinião: “Você sentiu que você e o seu time efetivamente utilizaram técnicas de revisão offline para testar seu projeto?”

Caracterização do efeito no processo de desenvolvimento

- Efetividade da aplicação de técnicas de revisão estão relacionadas com:
  - Porcentagem de testes operacionais bem sucedidos (nos times usando Cleanroom)
  - Completude da implementação (em todos os times)
- Não possuem relação com
  - Experiência profissional
  - Experiência em testes
- Obs: porque os times usando Cleanroom não podiam depender de métodos de teste, eles podem ter aplicado as técnicas de revisão offline mais efetivamente



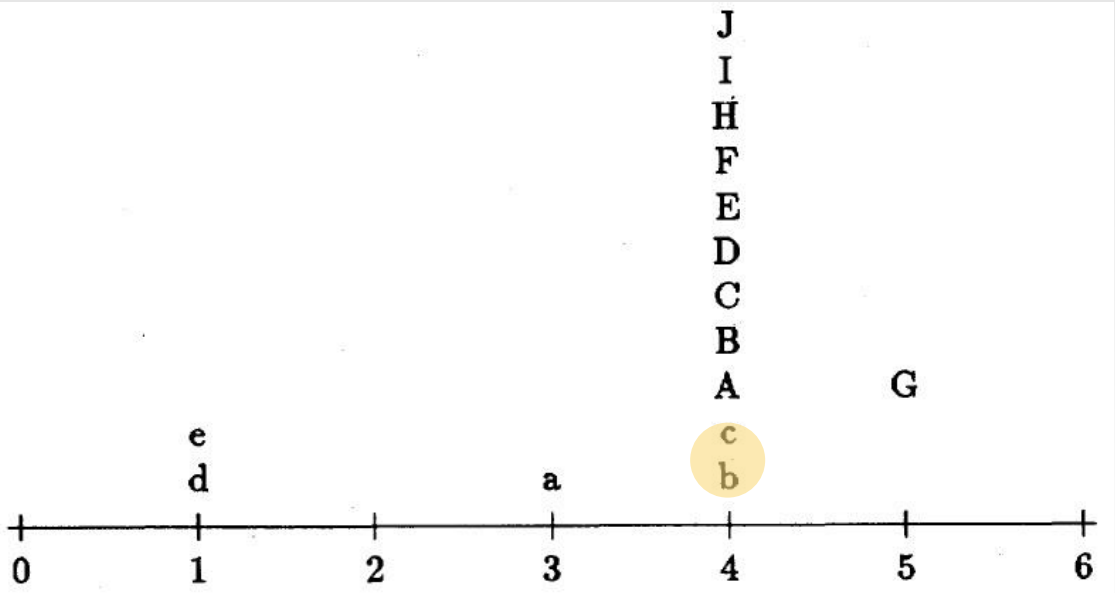
Mann-Whitney signif. = .089

Fig. 7. Connect time in hours during project development.[9]

- Desenvolvedores na abordagem Cleanroom gastaram menos tempo online e usaram menos recursos computacionais o que contribui para o papel reduzido do computador nesse modelo de desenvolvimento

Caracterização do efeito no processo de desenvolvimento

Enquanto todos os times utilizando a metodologia Cleanroom fizeram todas as entregas previstas, apenas um time utilizando a metodologia tradicional cumpriu o plano de entregas original

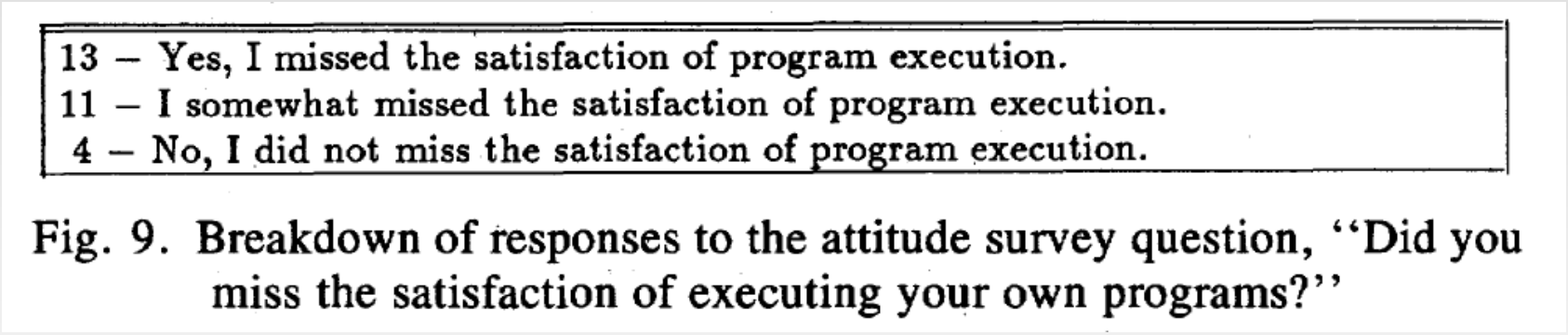


Mann-Whitney signif. = .006

Fig. 8. Number of system releases.

Caracterização do efeito nos desenvolvedores

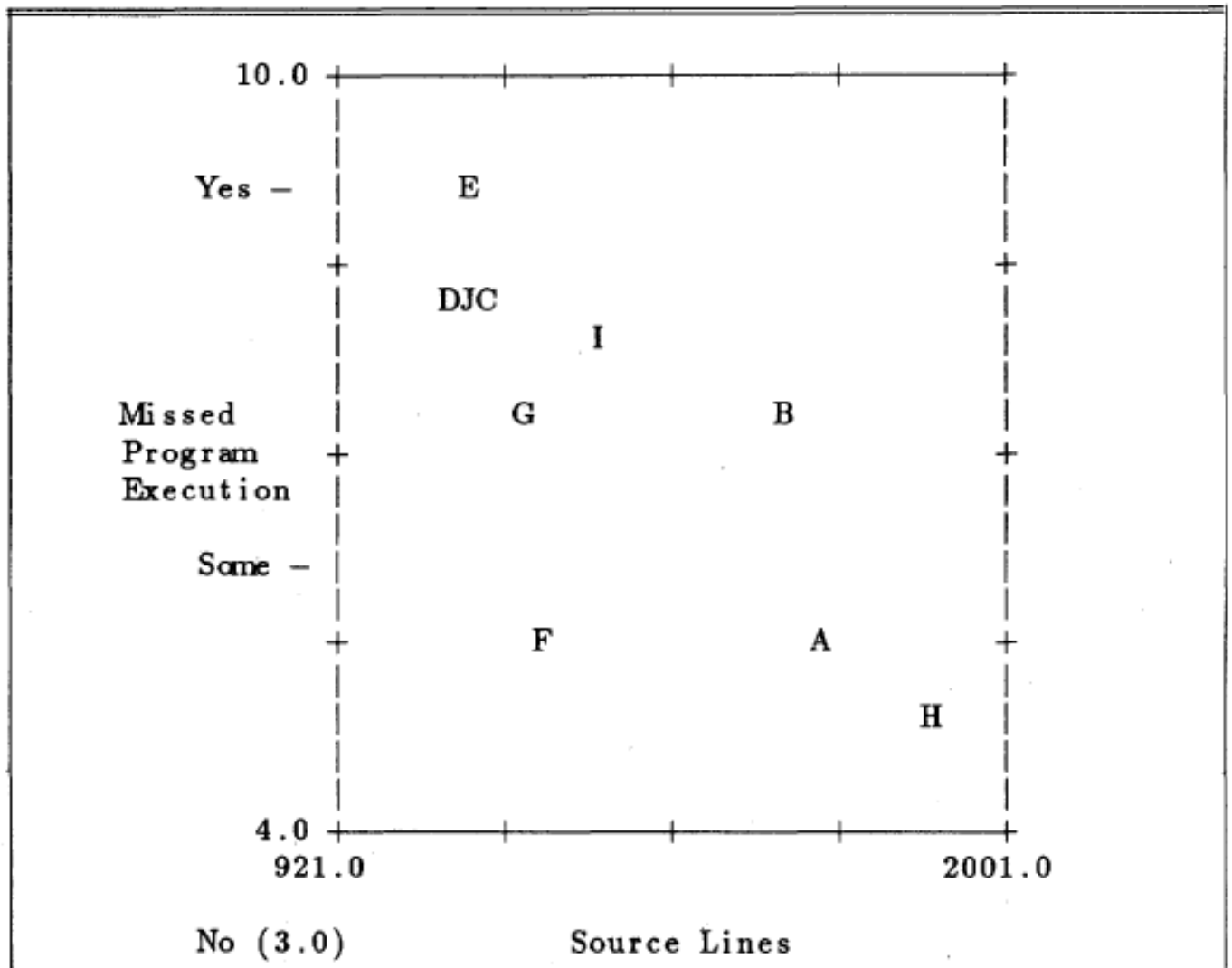
- Primeira pergunta



13	Yes, I missed the satisfaction of program execution.
11	I somewhat missed the satisfaction of program execution.
4	No, I did not miss the satisfaction of program execution.

Fig. 9. Breakdown of responses to the attitude survey question, "Did you miss the satisfaction of executing your own programs?"

- quase todos os indivíduos sentiram falta de algum aspecto da execução de um programa
  - não está relacionado com a experiência do indivíduo
  - não é maior com relação ao tamanho do programa



Spearman correlations: −.85 (signif. = .002) with source lines; −.70 (signif. = .03) with number separately compilable modules; −.57 (signif. = .09) with number procedures and functions.

Fig. 10. Relationship of program size versus missing program execution.

Caracterização do efeito nos desenvolvedores

2 – Yes, my style was substantially revised. 15 – I modified some of my tendencies. 11 – It did not affect my style at all.
Frequently mentioned responses include – kept design simple, attempted nothing fancy – kept readability of code in mind – already was a user of off-line review techniques – very careful scrutiny of code for potential mistakes – prepared for a larger range of inputs

Fig. 11. Breakdown of responses to the attitude survey question, "How was your design and coding style affected by not being able to test and debug?"

As a software development manager? 8 – Yes, at all times 14 – Yes, but only for certain projects 5 – Not at all
As a programmer? 4 – Yes, for all projects 18 – Yes, but not all the time 5 – Only if I had to 0 – I would leave if I had to

Fig. 12. Breakdown of responses to the attitude survey question, "Would you use Cleanroom again?" (One person did not respond to this question.)

Distinção entre os times

- Dois times perderam um membro (H e I)
  - H foi muito bem, diferentemente do I
- Os grupos do segundo grupo não foram divididos igualmente já que havia uma pessoa com muita experiência, essa pessoa formou um time de um homem só

Conclusões

- Cleanroom é uma abordagem que objetiva produzir software confiável através de especificação formal, desenvolvimento sem execução e testes estatísticos
- Foi conduzido um estudo replicado para verificar tal abordagem
- Os dados coletados sugerem as conclusões:
  - Maioria dos desenvolvedores é capaz de adotar Cleanroom de modo efetivo;
  - Produtos do Cleanroom satisfazem melhor os requisitos e testes em comparação com abordagem tradicional;
  - Código gerado pelo uso de Cleanroom possui mais comentários e menor complexidade de fluxo de controle;

Conclusões

- Conclusões (continuação)
  - Os melhores desenvolvedores Cleanroom obtiveram código fonte mais enxuto, com mais declarações IF e menos declarações WHILE e CASE;
  - Todas dez equipes utilizando Cleanroom tiveram entregas intermediárias; enquanto somente duas de cinco equipes de controle o fizeram;
  - 86% dos desenvolvedores que utilizaram Cleanroom sentiram falta da execução do código, mas sem impacto na qualidade do mesmo; e
  - 81% dos desenvolvedores que utilizaram Cleanroom utilizariam a abordagem novamente.

Conclusões

- Melhorias sugeridas
  - Permitir que desenvolvedores acompanhem eventuais execuções do código;
  - Permitir melhor entendimento das interfaces através de protótipos visuais, e assim melhor usabilidade dos produtos gerados; e
  - Expandir o processo de testes com maior cobertura dos requisitos.
- Cleanroom auxilia na obtenção de produtos de alta qualidade com confiabilidade operacional, facilitando a manutenção de código produzido