Prevendo Defeitos de Software I: Avaliação da Qualidade

Esta série de tutoriais sobre Defeitos de Software apresentará técnicas de prevenção desses defeitos usando os resultados obtidos na fase de testes.

Neste tutorial parte I apresentamos uma metodologia de avaliação da qualidade de software baseada no histórico de defeitos detectados na fase pré liberação. A avaliação fornece resultados para a criação de cenários consistentes, que serão usados na previsão de defeitos na fase pós liberação.

Jorge Moreira de Souza

Doutor em Informática (81) pelo Instituto Nacional Politécnico de Toulouse, França, Mestre (75) e Bacharel (71) em Engenharia Elétrica na PUC-RIO.

As principais áreas de interesse são Engenharia de Tráfego e Análise de Confiabilidade de Sistemas. Trabalha atualmente na FITec onde desenvolve trabalhos de análise/avaliação e revisões de projeto.

E-mail: jmdsouza@fitec.org.br

Duração estimada: 20 minutos

Publicado em: 13/03/2006

O desenvolvimento de software é marcado por preocupações como:

- Cumprimento do cronograma;
- Custos dentro do planejado;
- Confiabilidade e robustez do produto em campo.

Essas preocupações são motivadas por diversos insucessos de produtos de software que levaram à perda de mercado ou ao comprometimento da imagem em relação à qualidade e que tiveram origem no mau planejamento do desenvolvimento.

Em resposta a essas preocupações muito tem se feito em relação à organização e definição do processo de desenvolvimento de software e o controle de qualidade ao longo do desenvolvimento. Podemos citar o "*Capability Maturity Model (CMM)*" , Telcordia e a ISO 9001.

O **controle de qualidade** visa a detecção de desvios ao longo do processo de desenvolvimento. A análise dos desvios permite a correção pontual de rumo do desenvolvimento ou até mudanças no processo que beneficiarão também desenvolvimentos posteriores.

A **avaliação da confiabilidade** visa:

- **Avaliar a qualidade do software** por meio do controle de qualidade e de medidas como MTBF (*Mean Time Between Failures*), indisponibilidade, intensidade de falha, etc. Essas medidas são primordiais para sistema com requisito de alta confiabilidade e segurança (*safety*);
- **Prever o número de defeitos em campo** tanto quanto à severidade como ao volume para dar suporte aos contratos de garantia e estimar o esforço de manutenção.

Neste tutorial apresentaremos uma metodologia de avaliação da qualidade do software visando a previsão de defeitos e não a obtenção de medidas de confiabilidade. Uma proposta para a previsão do número de defeitos em campo será objeto de outro tutorial.

A metodologia de avaliação se baseia no histórico de defeitos detectados durante a fase de teste de sistema. Por **defeito** entendemos a detecção de uma operação do sistema que não está conforme a especificação e que necessita de uma correção do software para ser sanado.

A principal ferramenta usada para o **controle de qualidade** é o gráfico de controle bastante usado no controle de processos fabris de modo a garantir que o produto fabricado está dentro de determinadas especificações.

É uma ferramenta poderosa que já usamos no tutorial para análise de alarmes visando aumentar a disponibilidade da rede. O **gráfico de controle** é usado no controle da "fabricação" de software para garantir que a taxa de defeitos detectada ao longo do processo de desenvolvimento está dentro de limites especificados.

Não existe um modelo amplamente aceito para **avaliação da confiabilidade** de software tanto de avaliação da qualidade como de previsão de defeitos. O modelo mais utilizado e que adotaremos no tutorial é o de **crescimento de confiabilidade**. Os modelos são ditos de "crescimento de confiabilidade" porque supõem que o processo de detecção e correção de falhas é perfeito e consequentemente o sistema tende a crescer em confiabilidade ao longo do tempo.

Na prática isso não acontece e períodos de crescimento (diminuição da taxa de defeitos) e decrescimento (aumento da taxa de defeitos) de confiabilidade se sucedem ao longo do desenvolvimento ou de operação em campo. Mesmo com essas variações os modelos de crescimento de confiabilidade podem se aplicados desde que no momento da liberação (release) a tendência seja de crescimento de confiabilidade.

Vamos exemplificar o conceito de crescimento de confiabilidade usando os dados de defeitos detectados no Sistema Operacional *OpenBSD*. Esse sistema é um sistema aberto (*open source*) que disponibiliza na web o relatório de defeitos encontrados.

A figura 1 mostra a curva do número de defeitos detectados ao longo dos meses. A liberação da versão R 3.0 ocorreu no primeiro de dezembro de 2001. O que você diria sobre crescimento de confiabilidade baseado na figura 1?

Podemos concluir, com base no aumento do número de defeitos, que só há um crescimento de confiabilidade consistente após janeiro de 2002, quando o número de defeitos diminui ao longo dos meses. Até janeiro de 2002 o número de defeitos aumenta.

Podemos então concluir:

- Que não há crescimento de confiabilidade até janeiro de 2001;
- Que há crescimento consistente de confiabilidade a partir de janeiro de 2002.

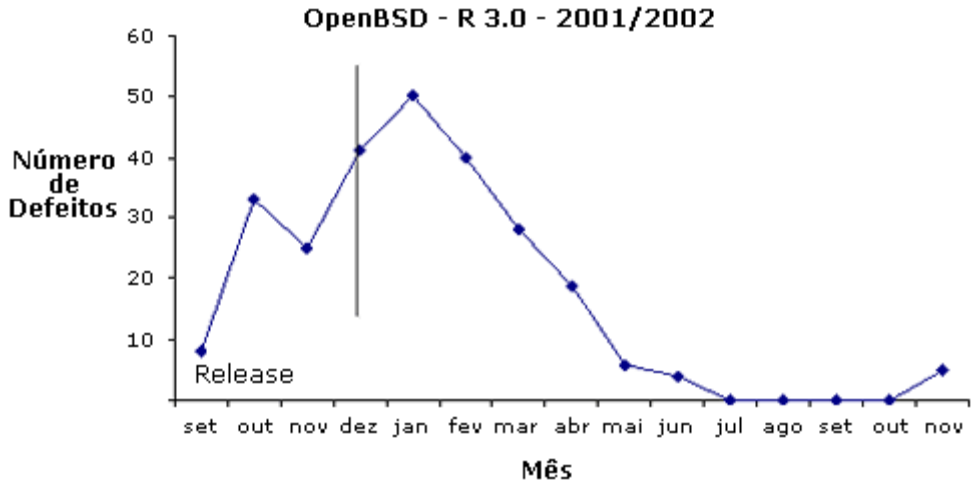

Figura 1: *OpenBSD* R 3.0 - Número de defeitos de software.

Nesse exemplo, há um período de decrescimento seguido de crescimento de confiabilidade. Isso é muito comum em vários casos de análise de defeitos de software.

O modelo teórico para representar esse comportamento deve ter também essa característica: decrescimento seguido de crescimento de confiabilidade. Porque a necessidade de um modelo teórico? Se quisermos fazer uma previsão usando o histórico de defeitos já detectados precisamos de um modelo.

Ou seja, o modelo teórico deve representar da melhor maneira possível o histórico de defeitos e, a partir daí, ser usado para prever o que deve acontecer em seguida. Vamos usar o modelo chamado *S-Shaped* [8] que explicaremos a seguir.

O modelo *S-Shaped*

As expressões do modelo S-Shaped:

$$H(t) = a[1 - (1 + bt)\exp(-bt)]$$
$$h(t) = ab^2 t\exp(-bt)$$

Onde:

- *H(t)* = a média acumulada de defeitos até o período t;
- *h(t) = dH(t)/dt* , a taxa média de manifestação de defeitos;
- *b* = a taxa média de ocorrência de um defeito;
- a = parâmetro que representa o número inicial de defeitos presentes no sistema.

O modelo *S-Shaped* supõe que há um período de decrescimento de confiabilidade antes de começar o período de crescimento de confiabilidade. Esse fenômeno é geralmente observado devido à possíveis instabilidades do sistema no início do período de teste ou de implantação. No modelo *S-Shaped* o **ponto de inflexão** (onde o sistema começa a crescer em confiabilidade) ocorre em *t = 1/b*.

Os parâmetros *a* e *b* do modelo *S-Shaped* devem ser estimados baseado no histórico de defeitos. Não discutiremos aqui os métodos de estimação. Veja a referência (J., Software Reliability Engineering, McGraw-Hill, 1999).

Existem outros modelos de crescimento de confiabilidade. Para quem quiser conferir, basta consultar as referências (J., Software Reliability Engineering, McGraw-Hill, 1999) e (Pham H., Nordmann L., Zhang X., "A General Imperfect-Software-Debugging Model with S-Shaped Fault-Detection Rate", *IEEE Trans Reliability*, VOL.48, NO.2, 1999 June).

Entendido o que é crescimento de confiabilidade, voltemos à análise do *OpenBSD* R 3.0.

Vamos aumentar a granularidade de observação e ver o que acontece no dia-a-dia, conforme mostrado na figura 2.

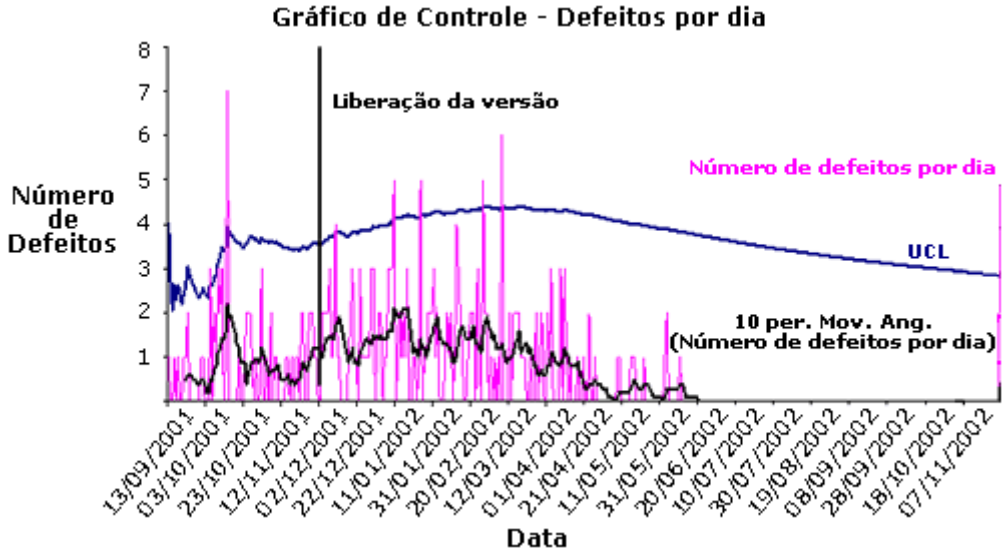

Figura 2: Número de defeitos por dia.

A figura 2 é um gráfico de controle onde é apresentado o limite superior de controle UCL (*Upper Control Limit*), o número de defeitos diários e a média móvel calculada sobre 10 dias.

Usaremos esse exemplo para apresentar a **metodologia de avaliação da qualidade**:

- Durante o período de teste, os pontos fora-de-controle (acima do UCL), devem ser analisados. A análise deve abordar pontos como: necessidade de ações de mudança no processo, necessidade de ações de complementação dos testes, chance do problema voltar a ocorrer mesmo com as ações tomadas, risco de comprometer outras funcionalidades devido às correções de software, etc;
- Análise de tendência da taxa de defeitos: um período continuado de crescimento da taxa de defeitos deve ser analisado da mesma forma como no item 1, mesmo que não ocorra pontos fora-de-controle.
- Analisando o *OpenBSD* R 3.0 no período pré liberação notamos:
- Um crescimento consistente da taxa de defeitos em outubro de 2001 que leva a um ponto fora-de-controle em meados desse mês;
- No período anterior a liberação (15 dias antes) há uma tendência consistente de crescimento da taxa de defeitos;
- Após a liberação a taxa de defeitos cresce. O crescimento consistente de confiabilidade só é observado em março de 2002.

De acordo com a metodologia, esses pontos deveriam ser analisados e as ações tomadas deveriam ser registradas.

De maneira geral, no momento de liberação podemos identificar quatro cenários possíveis:

- **Cenário 1, ideal:** o software apresenta um crescimento consistente de confiabilidade;

- **Cenário 2, quase ideal:** o software apresenta uma taxa de defeito crescente mas a avaliação é que trata-se um efeito temporário e que a tendência após a liberação será de crescimento de confiabilidade;
- **Cenário 3, crítico:** o software apresenta uma taxa de defeito decrescente e a avaliação é que trata-se um efeito temporário e que a tendência, após a liberação, será de decrescimento de confiabilidade durante um certo período;
- **Cenário 4, crítico:** o software apresenta uma taxa de defeito crescente e a avaliação é que trata-se um efeito consistente e que a tendência após a liberação será de decrescimento de confiabilidade durante um certo período.

O cenário ideal geralmente não ocorre devido às exigências contratuais, cronograma de entrega, time to market, etc. A análise durante o período de testes fornece elementos adicionais aos engenheiros de desenvolvimento e de teste para prever os possíveis cenários após a liberação.

Essa previsão é útil para dimensionar o esforço de manutenção, as cláusulas quantitativas do contrato de manutenção, o risco de situações críticas, o tempo para o software apresentar crescimento de confiabilidade, etc.

É aqui que o modelo de crescimento de confiabilidade tem sua utilidade: **ajudar a prever e quantificar o que vem depois.**

Como prever os defeitos de software usando os modelos de crescimento de confiabilidade?

Neste tutorial apresentaremos apenas os resultados usando o que já sabemos sobre o *OpenBSD* R 3.0. No próximo tutorial detalharemos nossa proposta de previsão baseada no histórico de defeitos, modelos de confiabilidade e construção de cenários.

Baseado nos resultados do OpenBSD R 3.0 consideramos um cenário quase ideal e dois críticos.

- **Cenário 1, quase ideal:** o crescimento da taxa de defeitos observada 15 dias antes da liberação é temporária. A tendência após a liberação é de crescimento de confiabilidade;
- **Cenário 2, crítico:** o crescimento da taxa de defeitos observada 15 dias antes da liberação é consistente e o software levará um mes para crescer em confiabilidade;
- **Cenário 3, crítico:** o crescimento da taxa de defeitos observada 15 dias antes da liberação é consistente e o software levará dois meses para crescer em confiabilidade.

A figura 3 mostra os resultados de previsão para os cenários descritos. A figura mostra o número acumulado de defeitos ao longo do tempo (em dias).

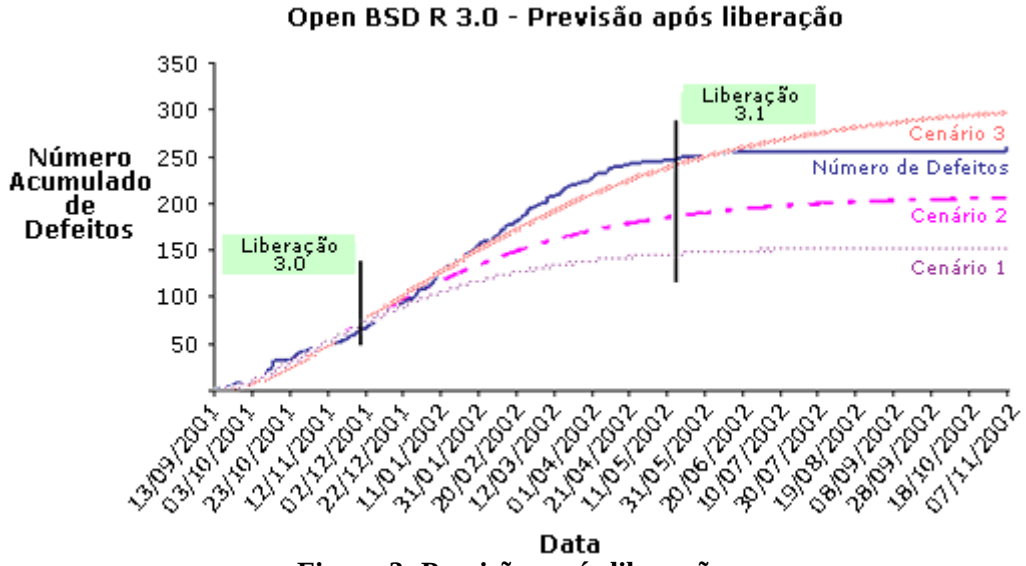

Figura 3: Previsões após liberação.

O cenário 3 é o que melhor descreve o que aconteceu. Nem sempre os engenheiros de desenvolvimento e teste chegam a um acordo sobre um único cenário.

O que se deve fazer nesse caso é uma análise de sensibilidade, do tipo "what-if", criando possíveis cenários para ajuda à tomada de decisão.

A avaliação da confiabilidade de software passa pela análise crítica do histórico de defeitos até o momento da liberação, que chamamos de **avaliação da qualidade**. No caso de sistemas com alta disponibilidade é exigido também a avaliação de indicadores como disponibilidade, MTTF, etc.

Baseado no resultado da análise de qualidade e em modelos de crescimento de confiabilidade de sofware é possível a criação de cenários que permitam a previsão do volume de defeitos pós liberação.

Neste tutorial mostramos a metodologia de avaliação da qualidade e como ela pode ajudar na criação de cenários. No próximo tutorial detalharemos nossa proposta de previsão de defeitos de software baseada no histórico de defeitos detectados durante os testes.

Referências

[1] http://www.sei.cmu.edu/cmm/

[2] Telcordia, The Analysis and Use of Software Reliability and Quality Data, SR-1547, Dec 2000.

[3] D.C.Montgomery, Introduction to Statistical Quality Control, John Wiley, 1007.

[4] J.Moreira de Souza, Aumentando a Disponibilidade da Rede por meio da Análise Estatística de Alarmes, http://www.teleco.com.br/tutoriais/tutorialdisponibilidade1/default.asp.

[5] W.A.Florac, A.D.Carleton, Measuring the Software Process: Statistical Process Control for Software Process Improvement, Addison-Wesley, SEI series in Software Engineering, 1999.

[6] Jacob A.L., Pillai S.K., Statistical Process Control to Improve Coding and Code Review, IEEE Software, May/June 2003.

[7] www.openbsd.org.

[8] Yamada S., Ohba M., "S-Shaped Reliability Growth Modeling for Software Error Prediction", *IEEE Trans Reliability*, VOL R-32, 1983 Dec

[9] J., Software Reliability Engineering, McGraw-Hill, 1999.

[10] Pham H., Nordmann L., Zhang X., "A General Imperfect-Software-Debugging Model with S-Shaped Fault-Detection Rate", *IEEE Trans Reliability*, VOL.48, NO.2, 1999 June.

1. Gráfico de controle é uma ferramenta poderosa que é usada para:

- Acompanhar os defeitos de fabricação.
- Garantir que um processo ou evento está dentro de determinados limites.
- Controlar o número de itens defeituosos.
- Explicar os pontos fora-da-curva.

2. Um software apresenta crescimento de confiabilidade quando:

- A taxa de defeito é nula.
- A taxa de defeito decresce.
- A taxa de defeito decresce consistentemente ao longo do tempo.
- Não há necessidade de continuar o teste.

3. Se o gráfico de controle apresenta pontos acima do limite superior (UCL), isso indica que:

- O software não está crescendo em confiabilidade.
- A taxa de defeito tende a crescer.
- Deve ser aumentado o número de testes.
- Que a taxa de defeitos está fora de limites especificados.

4. Os modelos de crescimento de confiabilidade de software são usados para:

- Avaliar a disponibilidade.
- Modelar o histórico de defeitos detectados e prever o comportamento futuro.
- Analisar os pontos fora-de-controle.
- Modelar os pontos for-de-controle.