Métricas de Software

Edson Orivaldo Lessa Junior

Importância das Métricas

- Permitem quantificar atributos de produtos e processos
- Facilitam tomadas de decisão baseadas em dados
- Apoiam a avaliação da qualidade, produtividade e desempenho
- Fundamentais para melhoria contínua

- é importante medir o código. Por mais que seja abstrato.
- uma tela pode ser muito complexa ou várias muito simples;
- medir o software não fácil.

- MÉTRICAS seria: medir numero de linhas, nº defeitos, tempo gasto em horas em um tarefas;
- tentativa em converter, avaliações por vezes subjetivas, em ordem objetiva.

- Códigos são extensos e complexo por natureza - o tipo do dado é complexo, uma linha é. Então como avaliar isso?

Códigos extensos e linha de código extenso:

- Software de seguro: Tal funcionalidade com 18000 linhas - não tem como manter isso em uma função!
- Melhoria continua - melhorar essas métricas ao longo do tempo: Todo o time deve estar envolvido de modo colaborativo. Identificar gargalos

Conceitos Básicos

ex: buscar um parâmetro - 100/1000 - não adianta.
Se 10000 até tantos erros. a cada 1000linhas, 10 erros.

Cria uma inicialmente e começa a diminuir cada vez mais.
- Quantos erros a cada quantas linhas?

- Medida: valor quantitativo (ex: 5000 linhas de código)
- Métrica: interpretação de medidas (ex: complexidade ciclomática)
  - qual o impacto de erros após uma implementação de um sistema? Para prever futuros cenários;
- Indicador: combinação de métricas para decisão (ex: produtividade da equipe)

sem contexto não pode se definir o grau de impacto. Preciso ver o relacionamento desse conteúdo com o fucionamento todo do programa;

Resultado da medida após a aplicação de uma fórmula: definir procedimento para que a medida faça sentido.
definir e quantificar atributos, com informações adicionais como MÉTRICA DE PRODUTO, DE PROCESSO, DE PROJETOS
Métrica de produto - Densidade de defeito;
Métricas de processo - tempo médio para solução de erros | Métricos de projeto - Esforço por funcionalidade entregue

interpretação da métrica - transforma os dados em informações úteis. Dá os rumos para a execução do projeto;

Tipos de Métricas

- Produto: tamanho, complexidade, qualidade de código
- Processo: tempo de execução, defeitos por fase
- Projeto: esforço, custo, prazo
- Uso: usabilidade, experiência do usuário

Cada tipo de métricas se referem a: o que medir, o por que medir e como medir?
Produto: tamanho do software, pontos de função, densidade de defeitos por 1000/linhas de código, cobertura de testes (automação de testes)

Processo: mede a eficiência e a qualidade dos processos realizados - tempo médio para corrigir defeitos, nº de retrabalhos em determinada fase, faixa de entrega no prazo, velocidade da equipe (no que se refere a Ponto de Função, Histórico de entrega nas Sprints);

Projeto: Dentro do projeto é mais fácil medir inicio, meio e fim. Envolve o gerenciamento do próprio projeto, planejamento, acompanhamento de cronogramas, orçamento para alocação de recursos (pessoas e software - equipamentos) - CUSTO POR FUNCIONALIDADE DE ENTREGA, PERCENTUAL DE ATRASO DE CADA FASE DO CRONOGRAMA, HORAS TRABALHADAS POR TAREFA "BURNDOWN CHART"
Uso: a ideia é forcar a UI, satisfação, retenção, tempo média da tela.

Framework HEART + GSM

- Foco: experiência do usuário
- Happiness: avaliações, feedback
- Engagement: frequência de uso
- Adoption: novos usuários/funções
- Retention: usuários recorrentes
- Task Success: taxa de conclusão

medir a UI de forma mais sistemática;
- H - mede a percepção subjetiva do usuário, satisfação na lojas, avaliações
- E - com que frequência o usuário interage com o sistema (Nº de sessões por dia, ações realizadas por login, tempo na seção)
_ A - quantos novos usuários começaram a usar o sistema (novos cadastrados por mês, ano)
- R - quanto tempo o usuário usa após instalar, ou quantos usuários voltam após cancelar, taxa de abandono
Task Succes - os usuários conseguem completar suas tarefas com facilidade?
Eficácia, Eficiência e Satisfação. Como um Mantenedor do Sistema consegue medir essas métricas? Logs do sistema previamente desenvolvidos para medir erros? números de cliques por função/por usuário? e gera um relatório depois ?

- aplicações consegue monitorar - Mapas de Calor

- tela aberta - e a pessoa não finalizou - EFICIENCIA

- como tirar a informação? Logs do sistema, ações e monitoramento de determinada funções. Cruzar logs com monitoramento.

Exemplo HEART Sistema Educacional

- Objetivo: Melhorar engajamento
- Sinal: Tempo por sessão > 5 min
- Métrica: Tempo médio de uso por sessão por aluno

quanto tempo os alunos estão na seção por mais de 5 minutos? Está engajado no conteúdo ou está com dificuldade de encontrar determinada funcionalidade.

a gamificaçao é proposital para gerar engajamento. Gera uma expectativa no usuário para que ele permaneça na aplicação.

Métricas para Requisitos

- Pontos de Função (Function Points): mede funcionalidades entregues
- Qualidade da Especificação: completude, consistência, rastreabilidade
- Fórmula FP: FP = Contagem total x [0,65 + 0,01 x Σ(Fi)]

precisão, usabilidade

- tentar medir o esforço, tentar compreender como será o software antes de desenvolvê-lo.
- Trazemos como ponto de partida, a documentação de requisitos.

Ex. Contar o que o sistema deve fazer, como se comportar e as respectivas restrições.
Avaliar a especificidade da qualidade software isso.

PF - medir o tamanho funcional viviveis para o usuario. Contas quantas funcionalidades. Com isso também é possível medir a performance de trabalho do time;

É usado também para faturamento. De acordo quantas funcionalidades, vai se pagando aos poucos para a empresa desenvolvedora a partir das funcionalidades entregues.

Exemplo Pontos de Função

- EIs: 3, EOs: 2, EQs: 2, ILFs: 1, EIFs: 4
- Contagem total: 50
- Fatores de ajuste: Σ(Fi) = 20
- FP = 50 x [0,65 + 0,01 x 20] = 50 x 0,85 = 42,5

Cálculo Esforço da Equipe:

- Se a equipe gasta 10 HORAS por Ponto de Função!
- Se 42,5 é a PF. Hora total será de 425 horas totais de trabalho.

PLANEJAR COM ISSO, CRONOGRAMA, RECURSOS!

Métricas para Projeto

- Arquitetura: complexidade estrutural, dependências
- Orientado a Objetos (CK):
- WMC: métodos ponderados por classe
- DIT: profundidade de herança
- RFC: resposta para classe

acomplamento do sistema

mostra a quantidade de métodos que podem ser acionados por acionamento de uma funcionalidade.

Exemplo CK Classe Usuario

- 10 métodos (WMC = 10) → quanto maior é a complexidade da classe, ela pode romper a barreira de complexidade máxima.
- Herança com 2 níveis (DIT = 2)
- RFC: 15 métodos acessíveis

a ideia é que uma classe não tenha muitas interações e dependências

Revisão de Métricas

- Objetivo: analisar resultados e promover melhorias
- Técnicas:
- Análise estatística
- Benchmarking
- Ferramentas de visualização de dados (ex: gráficos)
- Reuniões de retrospectiva

scrum, kanbam

Considerações Finais

- Métricas ajudam a melhorar qualidade de forma objetiva
- Devem ser interpretadas com contexto
- É importante revisar e adaptar as métricas periodicamente
- Foco sempre na melhoria contínua

BONS ESTUDOS