

- d) Em solução.
- e) Aguardando informação complementar.
- f) Aguardando aprovação.
- g) Aprovado/resolvido.

Em geral, o fato de um problema ter sido analisado e tratado não significa necessariamente que ele tenha sido resolvido. Assim, uma análise com fins de aprovação da resolução do problema deve ser feita ao final dos procedimentos de correção. Quando a solução final for aceita, necessariamente será gerada uma nova versão do artefato no sistema de controle de versões.

Como o acompanhamento de problemas deve ser feito por diferentes interessados, é natural que acabe sendo feito por um sistema informatizado. Um exemplo de software nacional gratuito para controle de problemas (manutenção) é o SIGMA¹⁷.

9.4.3 REGISTRO DE ARTEFATOS

O processo de acompanhamento de projeto e controle de problemas depende, de forma crucial, do sistema gerenciador de configuração ou controle de versões. O gerente deve se certificar de que todos os membros da equipe sigam corretamente a política de controle de versão estabelecida na empresa.

Uma política típica é que os desenvolvedores somente tenham acesso a um artefato quando liberado pelo seu proprietário. Eles podem trabalhar na cópia do artefato até realizar a tarefa a que se propuseram. Depois, devem submeter o artefato modificado ao controle de qualidade, em geral na forma de testes. Se aprovado, devem salvar uma nova versão do artefato no sistema de gerenciamento de configuração. Mais detalhes sobre esse processo serão apresentados no Capítulo 10.

9.5 Medição em Engenharia de Software

Um processo de gerência, para ser mais eficaz, precisa se basear em medições. Como saber se a tarefa não está sendo feita se não houver uma medida para chegar a essa conclusão? Como saber que a qualidade é inaceitável? De uma maneira ou de outra, o gerente de projeto vai acabar se envolvendo com a atividade de medição de software, na qual terá que aplicar uma ou mais métricas.

Inicialmente, alguns termos serão definidos para evitar confusão:

- a) *Medida*: valor obtido para alguma dimensão do software.
- b) *Métrica*: escala na qual os valores de uma medida são tomados.
- c) *Medição*: processo de obtenção de medidas.

O Software Engineering Institute (Mills E. E., 1988) indica que uma boa métrica precisa ter cinco qualidades:

- a) Ser *simples*, ou seja, ter uma definição curta e fácil de ser compreendida.
- b) Ser o mais *objetiva* possível, isto é, não depender de opiniões. Se duas pessoas avaliarem o mesmo produto usando essa métrica, deverão obter o mesmo resultado.
- c) Ser *facilmente obtida*, isto é, ter custo de medição razoavelmente baixo.
- d) Ser *válida*, isto é, indicar um valor que seja útil e efetivamente representativo da grandeza que se pretende medir.
- e) Ser *robusta*, isto é, pequenas mudanças no produto devem gerar mudanças proporcionais na medida. Apenas grandes mudanças no produto podem produzir grandes mudanças na medida.

Apesar disso, nem sempre as métricas apresentam todas essas qualidades. Um bom exemplo de métrica é a contagem de linhas de código (LOC; Seção 7.1). Desde que os padrões de contagem tenham sido estabelecidos (o que efetivamente conta e o que não conta), essa métrica deve produzir sempre o mesmo resultado. Já a contagem de pontos de função (Seção 7.3.3) a partir de um conjunto de requisitos poderá apresentar pequenas variações, dependendo da interpretação do analista sobre o que efetivamente se constitui em uma função individual do software e dos fatores técnicos. A avaliação dos fatores técnicos e ambientais, também, em pontos de caso de uso (Seção 7.5), é uma métrica

¹⁷Disponível em: <www.redeindustrial.com.br/site/empresa.aspx>. Acesso em: 21 jan. 2013.



bastante subjetiva. O método COCOMO II (Seção 7.3) tenta reduzir essa subjetividade nos multiplicadores de esforço e fatores de escala ao estabelecer uma série de padrões para atribuição de notas. Ainda assim, a métrica tem uma carga de subjetividade significativa.

A Seção 11.5 apresenta mais alguns detalhes sobre medição e métricas referentes à qualidade de software.

9.5.1 CLASSIFICAÇÕES DE MÉTRICAS

Existem várias classificações para métricas. Inicialmente, pode-se falar em métricas diretas, ou seja, aquelas que podem ser definidas e contadas de modo direto, sem necessidade de interpretação ou incerteza, como:

- a) Custo financeiro.
- b) Esforço em desenvolvedor-mês.
- c) Linhas de código (SLOC).
- d) Velocidade de execução em segundos.
- e) Memória em megabytes.
- f) Número de defeitos localizados (total ou relativo ao número de KSLOC).
- g) Complexidade ciclomática (Seção 13.4.1).

Outras métricas são indiretas e só podem ser determinadas a partir de uma definição operacional (Wazlawick, 2009), ou seja, um procedimento de medição que algumas vezes é *ad hoc*, e nem sempre será consenso. Exemplos de métricas indiretas são os seguintes:

- a) Funcionalidade.
- b) Qualidade.
- c) Complexidade.
- d) Eficácia.
- e) Confiabilidade.
- f) Manutenibilidade.
- g) Usabilidade.

Nota-se que as métricas mais importantes estão fortemente ligadas aos aspectos de qualidade do software (Seção 11.1).

Um aspecto que sempre deve ser lembrado é que só vale a pena coletar medições quando se tem em mente um propósito específico, pois a coleta de dados poderá acarretar trabalho extra antes, ao longo e depois do projeto de desenvolvimento. Assim, a coleta de medições deve ser encarada como um investimento de tempo e esforço com o objetivo de melhorar algum aspecto do produto ou do processo de desenvolvimento.

Do ponto de vista dos processos de gerência, pode ser interessante agrupar as métricas em termos de sua utilidade para o gerente. Assim, temos:

- a) **Métricas de produtividade:** custo, esforço (em AFP ou UCP) e KSLOC, que podem ser usadas para verificar o andamento do projeto e possíveis desvios.
- b) **Métricas de qualidade:** número de defeitos, eficiência, confiabilidade e capacidade de manutenção. São usadas para avaliar se o produto satisfaz aos critérios de aceitação para uso por parte do cliente e também critérios internos, que afetam a eficiência da equipe.
- c) **Métricas técnicas:** outros aspectos ligados ao produto, não necessariamente à qualidade ou à produtividade, mas a aspectos inerentes do sistema, como complexidade ciclomática, modularidade, paralelismo, distribuição etc.

As métricas podem ser *absolutas* ou *relativas*. Por exemplo, um sistema com cinco erros não é necessariamente pior do que um sistema com dois erros. Se o primeiro sistema tiver um milhão de linhas e o segundo tiver cinco mil linhas, então o segundo terá mais erros por linha do que o primeiro.

Assim, a medida relativa é frequentemente usada, em especial quando se deseja avaliar a qualidade do produto e do trabalho. Há pelo menos quatro formas relevantes de relativizar uma métrica:

- a) **Pelo tamanho:** divide-se o valor absoluto da métrica pelo número de linhas de código.
- b) **Pela funcionalidade:** divide-se o valor absoluto da métrica pelo número de pontos de função, pontos de caso de uso ou pontos de histórias.

- c) *Pelo tempo*: divide-se o valor absoluto pelo período de tempo. Por exemplo, número de defeitos detectados por mês.
- d) *Por esforço*: divide-se o valor absoluto pelo esforço despendido, geralmente em desenvolvedor-mês ou desenvolvedor-hora. Por exemplo, o número de linhas de código produzidas por desenvolvedor-mês.

Talvez o trabalho mais formal na área de métricas esteja relacionado à medição da complexidade do software, o que é útil para a área de testes (Capítulo 13). Outro tipo de métrica que recebeu muita atenção é a métrica de qualidade ligada ao produto (Capítulo 11), embora às vezes elas sejam contraditórias entre si. Por exemplo, aumentar a flexibilidade (desejável) pode diminuir a eficiência de tempo (indesejável).

Entretanto, algumas métricas de qualidade podem ser especialmente úteis ao gerente de projeto, pois melhorar suas medições será quase sempre muito salutar:

- a) *Métricas de defeitos*: o número de defeitos em um produto de software deveria ser uma métrica objetivamente contável. Porém, encontrar defeitos não é uma tarefa trivial. Assim, normalmente, a medição de defeitos é feita por uma das seguintes técnicas:
 - Número de alterações de *design* ou código.
 - Número de erros detectados nas inspeções de código (Seção 11.4.2).
 - Número de erros detectados nos testes de programa (preferencialmente pela equipe de teste, mas também, possivelmente, pelos usuários).
- b) *Métricas de confiabilidade*: uma vez que se tenha uma medida nominal dos defeitos encontrados em um produto de software, sua confiabilidade pode ser calculada para determinado período de tempo. Se um sistema complexo apresenta certa taxa de falhas ao longo de um mês ou de um ano, pelas Leis de Lehman (Seção 14.1), pode-se esperar que continue exibindo esse comportamento ao longo dos meses ou anos seguintes. Essas medidas podem ser tomadas tanto durante o processo de desenvolvimento quanto durante o período de operação pós-desenvolvimento do sistema.
- c) *Métricas de manutenibilidade*: embora a manutenibilidade seja, a princípio, uma métrica subjetiva, existem estudos que mostram que a complexidade do produto (a complexidade ciclomática, por exemplo) afeta o esforço necessário para encontrar e reparar defeitos no software. Assim, existe uma relação direta entre o número de defeitos encontrados e a complexidade do software com o esforço necessário para fazer manutenções. Novamente, essas atividades de manutenção podem ser tanto aquelas que ocorrem durante a operação do software quanto as modificações que ocorrem durante o desenvolvimento.

Deve-se ficar atento, porém, para o fato de que essas métricas precisam ser sempre interpretadas em seu contexto. Por exemplo, a detecção de muitos defeitos no software pode significar tanto que a atividade de teste está sendo bem conduzida quanto que as atividades de programação estão sendo mal conduzidas. Mas defeitos detectados por usuários durante o uso do sistema costumam ser péssimas notícias.

9.5.2 PLANEJAMENTO DE UM PROGRAMA DE MÉTRICAS

Para que um gerente possa utilizar métricas para dirigir suas atividades, ele deve primeiro implantar um *programa de métricas*. Segundo o SEI (Mills E. E., 1988), essa atividade deve ser muito bem planejada.

Inicialmente, devem-se estabelecer os objetivos do programa de métricas: Quais falhas ele vai corrigir? Quais aspectos ele vai tentar melhorar? Em seguida, devem ser estimados os custos do programa e deve ser obtido o apoio da gerência superior, pois esse tipo de ação normalmente precisa de um suporte razoável para ocorrer. Em relação aos custos, estes podem ser divididos em custos iniciais de implantação e custos de manutenção do programa.

Em função dos objetivos e do aporte financeiro inicialmente comprometido, deve-se escolher o conjunto de métricas a serem implementadas e o modelo de avaliação (por exemplo, para estimar o tamanho de um projeto, podem-se usar os modelos CII, Pontos de Função ou Pontos de Caso de Uso, cada qual com suas peculiaridades e custos específicos). Durante o processo de seleção das métricas e modelos devem-se observar os seguintes pontos:

- a) *Habilidade projetada para satisfazer objetivos*: as métricas e modelos disponíveis devem ser analisados e comparados em relação a sua capacidade de satisfazer os objetivos do programa de métricas.

b) *Dados e custos necessários estimados*: os modelos que são capazes de satisfazer os objetivos devem ser comparados em função de seu custo de implantação e manutenção e da quantidade e variedade de dados necessários para funcionar. Normalmente, os modelos mais econômicos são preferíveis.

Uma vez que o modelo tenha sido escolhido, deve-se identificar e refinar os dados que serão coletados. Apenas dados que possam satisfazer a algum objetivo imediato ou de longo prazo devem ser coletados. Deve-se evitar coletar quaisquer dados só porque estão disponíveis, pois o excesso de dados pode causar confusão quando se tenta fazer algum tipo de análise. Para executar essa atividade, deve-se observar o seguinte:

- a) *Especificidade dos dados*: os dados devem ser definidos e obtidos ao longo de todo o ciclo de desenvolvimento do software. Preferencialmente, deve-se identificar *quando* os dados foram obtidos. Isso permite analisar diferentes significados em diferentes fases ou para diferentes atividades do processo de desenvolvimento.
- b) *Procedimentos de obtenção de dados*: uma vez que os dados específicos tenham sido definidos, os procedimentos para sua coleta e as pessoas responsáveis devem ser identificados.
- c) *Manutenção do banco de dados*: uma vez que o banco de dados de medições passará a ser um importante patrimônio da empresa, os recursos para sua perfeita manutenibilidade devem ser definidos e destinados.
- d) *Previsões refinadas de esforço e custo*: com as informações obtidas nos itens anteriores, deve ser possível obter uma estimativa bem mais realista de custo e esforço para a implantação do programa de métricas.

Assumindo que todos os passos anteriores foram executados com sucesso e os custos são aceitáveis, o programa de métricas pode ser iniciado. Os seguintes itens ainda precisam ser enfatizados nessa fase:

- a) *Esclarecimento de uso*: os objetivos do programa de métricas devem ficar claros desde o início. Mas, no momento de iniciar seu uso, pode ser importante relembrá-los a toda a equipe. As pessoas devem ser informadas sobre as medidas que serão obtidas e o uso que será feito delas. Muito cuidado deve ser tomado especialmente se as medidas forem utilizadas para avaliação de membros da equipe. Sem uma ampla discussão e aceitação das métricas, essas iniciativas poderão dar origem a estresse e até sabotagens ao programa.
- b) *Pessoal responsável*: os responsáveis pela coleta, manutenção e interpretação dos dados devem ser definidos e informados. Deve-se lembrar que essa atividade deve ser executada continuamente ao longo de qualquer projeto, pois muitos dados poderão não ser mais obtidos depois que ele terminar.

Para que o programa de métrica tenha sucesso, ele deve ser continuamente usado, avaliado e reajustado. Os seguintes aspectos são relevantes:

- a) *Avaliação de resultados*: os resultados deveriam ser cuidadosamente resumidos e comparados com a realidade subjetivamente observada. Por vezes, algum desvio em relação à percepção da equipe pode ser resultado de erros no processo de obtenção dos dados.
- b) *Ajuste do modelo*: muitos modelos de medição exigem que determinadas constantes sejam calibradas ao longo de seu uso (por exemplo, CII). Assim, esses procedimentos de calibração devem ser executados sempre que o modelo assim o exigir.

Finalmente, convém lembrar que, em termos de capacidade de processos e maturidade de empresas (CMMI), é absolutamente necessário que exista um plano de métrica e que ele seja efetivamente usado para melhorar aspectos dos processos da empresa. Os níveis 4 e 5 de maturidade do CMMI só são atingidos caso exista um plano de métricas (Seção 12.3).

9.6 Revisão e Avaliação

Nos métodos ágeis, como Scrum e XP, a revisão e a avaliação de projeto são previstas e ocorrem informalmente nas reuniões diárias em pé e com mais formalidade nas reuniões de fechamento de iteração.

Outros métodos poderão definir reuniões de revisão de projeto, de forma periódica, seja ao final de uma iteração ou fase, seja a qualquer momento.

É importante que, para serem efetivas, as reuniões sejam planejadas com antecedência. As pessoas envolvidas devem ser comunicadas, o local adequado deve ser reservado e, o que é mais importante, os objetivos da reunião devem ser claramente transmitidos. Quando se fala em objetivos, deve-se entender isso no sentido de definir quais